

INSTITUTO TECNOLÓGICO SUPERIOR DE CALKINÍ

“DESARROLLO DE UN SISTEMA DE INTELIGENCIA DE NEGOCIOS PARA LA IDENTIFICACIÓN Y MEJORA DE PERFILES EMPRENDEDORES: UN ESTUDIO DE CASO EN PERFILES DE EGRESO UNIVERSITARIO”

TESIS PROFESIONAL

**PARA OBTENER EL TÍTULO DE:
INGENIERÍA EN SISTEMAS COMPUTACIONALES**

**QUE PRESENTA:
FERNANDA YAZBETH CHABLE COCON**

**DIRECTOR Y CO-DIRECTOR:
DR. GONZALO MIGUEL QUETZ AGUIRRE
DRA. MARLENE MÉNDEZ MORENO**

Calkiní, Campeche, México 2025

INSTITUTO TECNOLÓGICO SUPERIOR DE CALKINÍ

**“DESARROLLO DE UN SISTEMA DE INTELIGENCIA DE NEGOCIOS PARA LA
IDENTIFICACIÓN Y MEJORA DE PERFILES EMPRENDEDORES: UN ESTUDIO
DE CASO EN PERFILES DE EGRESO UNIVERSITARIO”**

TESIS PROFESIONAL

**PARA OBTENER EL TÍTULO DE:
INGENIERÍA EN SISTEMAS COMPUTACIONALES**

**QUE PRESENTA:
FERNANDA YAZBETH CHABLE COCON**

**DIRECTOR Y CO-DIRECTOR:
DR. GONZALO MIGUEL QUETZ AGUIRRE
DRA. MARLENE MÉNDEZ MORENO**

Calkiní, Campeche, México 2025



Educación
Secretaría de Educación Pública



Instituto Tecnológico Superior de Calkiní
Subdirección de Planeación y Desarrollo

Instituto Tecnológico Superior de Calkiní
Departamento de Administración y
Apoyo al Estudiante

Calkiní, Campeche 07/03/2025
P/03/25/650

INGENIERÍA EN SISTEMAS COMPUTACIONALES

CONSTANCIA DE NO INCONVENIENCIA

MODALIDAD: TESIS

C. FERNANDA YAZBETH CHABLE COCON 7500
PASANTE DE LA CARRERA DE
INGENIERÍA EN SISTEMAS COMPUTACIONALES.
PRESENTE

Con relación a lo establecido en el Reglamento de Estudios de Licenciatura, capítulo VII, fracción I, vigente en este Instituto, y una vez emitido el fallo favorable por el director DR. GONZALO MIGUEL QUETZ AGUIRRE, y el co-director DRA. MARLENE MENDEZ MORENO, respectivamente, este Instituto consta de no tener inconveniencia para la realización del acto protocolario de titulación integral de su trabajo profesional titulado: **""DESARROLLO DE UN SISTEMA DE INTELIGENCIA DE NEGOCIOS PARA LA IDENTIFICACION Y MEJORA DE PERFILES EMPRENDEDORES: UN ESTUDIO DE CASO EN PERFILES DE EGRESO UNIVERSITARIO""**.

ATENTAMENTE

Excelencia en Educación Tecnológica®
Sabiduría, Fortaleza de Nuestra Cultura®



INSTITUTO TECNOLÓGICO SUPERIOR
DE CALKINÍ
"DEPARTAMENTO DE ADMINISTRACIÓN
ESCOLAR Y APOYO A
ESTUDIANTES"
CLAVE: 04MSU0013P
CALKINÍ, CAMPECHE, MÉXICO.

Dr. Jorge Alfonso Hau Puc.
Departamento de Administración y
Apoyo al Estudiante.

C.c.p. Archivo



2025
Año de
La Mujer
Indígena

Av. Ah Canul S/N por Carretera Federal, Calkiní, Campeche,
C.P. 24900 Tel. 996-8134870,
e-mail: se_dcalkini@tecnm.mx | itescam.edu.mx



Dedicatoria.

A mi madre Magali Cocon por todo el apoyo que me dio a lo largo de la carrera, por todo el amor que me ha brindado a lo largo de mi vida, por siempre estar ahí cuando más la necesite por nunca dejarme sola con todos los obstáculos y siempre escucharme, fue mi principal motivo de este logro.

A mi padre Edgar Chable por el apoyo, la sabiduría y siempre impulsarme a seguir adelante y nunca rendirme a pesar de todos los obstáculos.

A mis hermanos con los que he compartido todos mis logros, alegrías, aventuras. Gracias por estar ahí animándome para seguir, siempre con una sonrisa y por supuesto hacerme reír cuando más lo necesite.

Al amor de mi vida por todo su apoyo incondicional que me ha dado por el cariño, amor, compañía que me dio cuando más lo necesite, por desvelarse y acompañarme en las noches de trabajos pesados, por siempre estar ahí echándome porras para no rendirme, por celebrar todos mis logros hasta los más pequeños.

A Dios y a mis ángeles de la guarda por ser la luz de mi vida, guiar mi camino por darme fuerza en mis días difíciles para superar los obstáculos y nunca rendirme.

Para ustedes les dedico todos mis logros y este proyecto. Los amo y gracias por todo el apoyo dado...

"El futuro pertenece a quienes creen en la belleza de sus sueños."

Eleanor Roosevelt

Agradecimientos.

En primer lugar, a mí misma por nunca rendirme a pesar de todos los problemas, obstáculos, las largas noches de desvelo y por mantener mi determinación para alcanzar este logro, por nunca dudar de mis capacidades a pesar de los prejuicios de la gente, por siempre mantenerme firme y nunca derrumbarme ante los múltiples desafíos.

Agradezco a mis padres por todo su apoyo incondicional que me han brindado a lo largo de la carrera por siempre estar ahí cuando los necesite. A mi madre por siempre animarme con amor para no rendirme y siempre lograr mis objetivos y metas. Gracias por brindarme de su apoyo económico y material, que gracias a eso pude concentrarme en mis estudios y conseguir este logro.

A mi compañero de vida por su apoyo incondicional, paciencia y amor. Gracias por estar a mi lado en cada paso de este viaje, por creer en mí cuando las fuerzas flaqueaban y por echarme porras siempre que las necesitaba.

Agradezco a mis asesores el DR. Gonzalo Miguel Quetz Aguirre y la Dra. Marlene Méndez Moreno por guiarme a lo largo de todo este proyecto, por los consejos, las recomendaciones y observaciones.

A mis docentes que han sido parte de mi trayectoria universitaria. Gracias por brindarme conocimientos valiosos y por exigirme siempre lo mejor de mí, enseñándome que la disciplina y el esfuerzo son clave para alcanzar el éxito.

Índice.

<i>Dedicatoria.</i>	4
<i>Agradecimientos.</i>	5
<i>Índice.</i>	6
<i>Índice Figuras.</i>	10
<i>Índice Tablas.</i>	21
<i>Resumen.</i>	22
Capítulo 1	23
1.1 INTRODUCCIÓN	23
1.2 ANTECEDENTES. Y/O MARCO TEÓRICO	24
1.2.1 PLANTEAMIENTO	26
1.2.2 JUSTIFICACIÓN	28
1.3 OBJETIVOS GENERALES Y ESPECIFICOS.	28
1.4 ALCANCES Y LIMITACIONES.	30
Capítulo 2. MARCO TEORICO	31
2.1. EMPRENDIMIENTO	31
2.1.1. Definición de Emprendimiento	31
2.1.2. Tipos de Emprendimiento	31
2.1.3. Fases del Proceso Emprendedor	31
2.1.4. Retos y Oportunidades en el Emprendimiento	32
2.2. INTELIGENCIA DE NEGOCIOS	33
2.2.1. Definición de Inteligencia de Negocios	33
2.2.2. Componentes de la Inteligencia de Negocios	33
2.2.3. Herramientas de Inteligencia de Negocios	33
2.2.4. Aplicaciones de la Inteligencia de Negocios en el Emprendimiento	34
2.3. LAS CARACTERÍSTICAS EMPRENDEDORAS PERSONALES (CEPS)	34
2.3.1. Definición de CEPs	34
2.3.2. Principales Características Emprendedoras	34
2.3.3. Evaluación de CEPs	36

2.3.4. Impacto de las CEPs en el Éxito Emprendedor.....	38
2.4. METODOLOGÍA SCRUM	39
2.4.1. Introducción a Scrum.....	39
2.4.2. Componentes de Scrum.....	39
2.4.3. Proceso de Implementación de Scrum.....	40
2.4.4. Ventajas y Desafíos de Scrum.....	41
2.5. ESCALA LINKERT	42
2.5.1. Estructura y funcionamiento.....	43
2.5.2. Aplicaciones en investigación.	43
2.5.3. Ventajas y desventajas.	43
2.5.4. Métodos de análisis de datos.....	44
2.6. PERFIL DE EGRESO	44
2.6.1. Importancia del Perfil de Egreso.	44
2.6.2. Componentes del Perfil de Egreso.	44
2.6.3. Diseño del Perfil de Egreso en México.....	45
2.6.4. Relación con el Mercado Laboral.	45
2.6.5. Evaluación y Seguimiento del Perfil de Egreso.....	45
2.7. ESTUDIO LONGITUDINAL	46
2.7.1. Características del Estudio longitudinal.....	46
2.7.2. Tipos de estudios longitudinales.....	46
2.7.3. Aplicaciones del Estudio Longitudinal.	46
2.7.4. Ventajas y Desafíos del Estudio Longitudinal.	47
2.7.5. Estudios Longitudinales en México.....	47
Capítulo 3.....	48
3.1 MATERIALES Y MÉTODOS.	48
3.1.1 RECURSOS SOFTWARE.....	48
3.2 DIAGRAMA DE BLOQUES DE LAS ETAPAS DEL PROYECTO	50
3.3 IMPLEMENTACIÓN DE LA METODOLOGÍA DEL PROYECTO	51
3.4 REQUERIMIENTOS	52
3.4.1 REQUERIMIENTOS FUNCIONALES.....	53
3.4.2 REQUERIMIENTOS NO FUNCIONALES.....	56
3.5. PROPUESTA PRELIMINAR DE DISEÑO Y FUNCIONALIDADES	56

MODULO DE BIENVENIDA.....	56
MÓDULO DE REGISTRO	63
MÓDULO DE INICIO DE SESIÓN	65
MÓDULO DE PERFIL DE USUARIO.....	65
MÓDULO DE GESTIÓN DE CUESTIONARIOS.....	68
MÓDULO DE GESTIÓN DE GRUPOS.....	74
MÓDULO DASHBOARD	76
MÓDULO DE ANÁLISIS Y REPORTES.....	84
3.6. CASOS DE USO.....	87
3.6.1 DIAGRAMA CASO DE USO PARA INICIAR SESION	87
3.6.2 DIAGRAMA CASO DE USO PARA EL ESTUDIANTE.....	87
3.6.3 DIAGRAMA CASO DE USO PARA EL MAESTRO.....	89
3.6.4 DIAGRAMA CASO DE USO PARA EL ADMINISTRADOR.....	89
3.7. MODELO RELACIONAL	90
3.8. IMPLEMENTACIÓN Y FUNCIONAMIENTO DEL SISTEMA DE INTELIGENCIA DE NEGOCIOS.....	91
3.9. PRUEBAS EXPLORATORIAS.....	120
3.10. CASO DE PRUEBAS DE USABILIDAD.	122
3.10.1. Módulo de Bienvenida.....	122
3.10.2. Módulo de Inicio de Sesión.	123
3.10.3. Módulo de Perfil de Maestro.	126
3.10.4. Módulo de Estadísticas y Comparación de Grupos.	129
3.10.5. Módulo de Reportes.....	130
3.10.6. Módulo de Gestión de Usuarios.....	130
3.10.7. Módulo de Perfil de Administrador.....	131
3.11. PRUEBAS DE ACEPTACIÓN.	134
3.11.1. Módulo de Bienvenida.....	134
Capítulo 4. RESULTADOS Y DISCUSION.	137
Capítulo 5. CONCLUSIONES Y RECOMENDACIONES.....	175
5.1 Conclusiones	175
5.2 Limitaciones y propuestas de mejora.....	176
Referencias Bibliográficas.....	177
ANEXOS.	182

Pruebas de Aceptación.....	182
Pruebas de Usabilidad	183
Backend.....	184
Migraciones	185
Modelos.....	207
Controladores.....	226
Routes	280
Frontend.....	283
Componentes.....	285
Models.....	325
Services	331
Validators	337
Routes	340

Índice Figuras.

Ilustración 3. 1 Diagrama de etapas del proyecto	51
Ilustración 3. 2 Diseño del menú principal de la plataforma	57
Ilustración 3. 3 Diseño del encabezado de la página principal	58
Ilustración 3. 4 Diseño del carrusel de fotos y objetivos	58
Ilustración 3. 5 Diseño de advertencia y pie de pagina	59
Ilustración 3. 6 Diseño de la ventana Quienes somos	60
Ilustración 3. 7 Diseño de la ventaja misión y visión del proyecto	60
Ilustración 3. 8 Diseño de la ventana Equipo	61
Ilustración 3. 9 Diseño de la ventana para contactos	62
Ilustración 3. 10 Diseño para la ventana de bloqueo	63
Ilustración 3. 11 Diseño del registro de usuario	64
Ilustración 3. 12 Diseño del inicio de sesión	65
Ilustración 3. 13 Diseño del dashboard de estudiante	66
Ilustración 3. 14 Diseño del perfil del maestro	67
Ilustración 3. 15 Diseño del perfil administrativo	68
Ilustración 3. 16 Diseño preliminar de la edición del cuestionario	69
Ilustración 3. 17 Mensaje de activación del cuestionario	70
Ilustración 3. 18 Diseño de la información del grupo (código de acceso)	70
Ilustración 3. 19 Diseño de la vista de las instrucciones del cuestionario	71
Ilustración 3. 20 Diseño del cuestionario	72
Ilustración 3. 21 Diseño de resultados del cuestionario	73
Ilustración 3. 22 Diseño de vista de los cuestionarios pendientes	74
Ilustración 3. 23 Diseño de módulo de gestión de grupos	74
Ilustración 3. 24 Diseño del módulo para la creación de los grupos	75
Ilustración 3. 25 Mensaje de grupo creado exitosamente	75
Ilustración 3. 26 Diseño de vista para la búsqueda de grupos	76
Ilustración 3. 27 Diseño de mensaje de éxito al unirte a un grupo	76
Ilustración 3. 28 Diseño de vista preliminar para el dashboard administrativo	77
Ilustración 3. 29 Diseño de vista preliminar para el dashboard para maestros	77
Ilustración 3. 30 Diseño de vista preliminar para el dashboard para alumnos	78
Ilustración 3. 31 Visualización de los cuestionarios resueltos (Alumno)	79
Ilustración 3. 32 Diseño de la vista de los grupos	80
Ilustración 3. 33 Diseño de la vista para la gestión de alumnos de un grupo	81
Ilustración 3. 34 Diseño de cuestionarios aplicados para el administrador	82
Ilustración 3. 35 Diseño preliminar para la administración de usuarios (Administradores)	83
Ilustración 3. 36 Diseño preliminar para la lista de usuarios (Dashboard de administradores)	83
Ilustración 3. 37 Diseño preliminar de la vista para resultados	84
Ilustración 3. 38 Diseño preliminar para resultados	85
Ilustración 3. 39 Diseño preliminar para exportaciones de resultados grupales	86
Ilustración 3. 40 Diseño preliminar para exportaciones de resultados individuales	86
Ilustración 3. 41 Diagrama de uso para inicio sesión	87

Ilustración 3. 42 Diagrama de uso para el estudiante	88
Ilustración 3. 43 Diagrama de uso para el maestro.....	89
Ilustración 3. 44 Diagrama de uso para el administrador	90
Ilustración 3. 45 Modelo Relacional	91
Ilustración 3. 46 Vista principal del usuario.....	92
Ilustración 3. 47 Registro para los alumnos	93
Ilustración 3. 48 Inicio de sesión para los usuarios	93
Ilustración 3. 49 Dashboard de alumnos	94
Ilustración 3. 50 Visualización de grupos del alumno	94
Ilustración 3. 51 Búsqueda de grupos.....	95
Ilustración 3. 52 Información del grupo encontrado	95
Ilustración 3. 53 Mensaje de éxito de ingreso al grupo	95
Ilustración 3. 54 Cuestionarios resueltos de un grupo (si no hay cuestionarios).....	96
Ilustración 3. 55 Cuestionarios resueltos (Si ya respondió algún cuestionario).....	96
Ilustración 3. 56 Cuestionarios pendientes del alumno	97
Ilustración 3. 57 Visualización del cuestionario	97
Ilustración 3. 58 Visualización del cuestionario	98
Ilustración 3. 59 Información del alumno	98
Ilustración 3. 60 Tabla de puntajes.....	99
Ilustración 3. 61 Grafico de puntajes	99
Ilustración 3. 62 Dashboard para maestros	100
Ilustración 3. 63 Visualización de grupos del maestro	100
Ilustración 3. 64 Formulario para crear el grupo	101
Ilustración 3. 65 Mensaje de éxito de grupo creado	101
Ilustración 3. 66 Información del grupo	101
Ilustración 3. 67 Información de los integrantes del grupo	102
Ilustración 3. 68 Búsqueda del alumno.....	102
Ilustración 3. 69 Mensaje de éxito de alumno agregado al grupo	103
Ilustración 3. 70 Lista de cuestionarios aplicados.....	103
Ilustración 3. 71 Visualización de publicación de cuestionario existente a un grupo	104
Ilustración 3. 72 Formulario para publicar un cuestionario existente	104
Ilustración 3. 73 Visualización del cuestionario a publicar (Cuestionario seleccionado)	105
Ilustración 3. 74 Mensaje cuestionario publicado	105
Ilustración 3. 75 Información del cuestionario aplicado en el grupo	106
Ilustración 3. 76 Administración de alumnos del maestro	106
Ilustración 3. 77 Visualización de cuestionarios resueltos del alumno	106
Ilustración 3. 78 Información del alumno	107
Ilustración 3. 79 Tabla de puntajes por característica	107
Ilustración 3. 80 Grafico de resultados	108
Ilustración 3. 81 Dashboard del administrador.....	108
Ilustración 3. 82 Apartado de países	109
Ilustración 3. 83 Agregar país.....	109
Ilustración 3. 84 Instituciones por país	110
Ilustración 3. 85 Visualización de maestros y alumnos por institución.....	110

Ilustración 3. 86 Lista de docentes por institución	111
Ilustración 3. 87 Formulario de registro del docente	111
Ilustración 3. 88 Lista de grupos del docente	112
Ilustración 3. 89 Lista de integrantes del grupo	112
Ilustración 3. 90 Lista de alumnos de la institución	113
Ilustración 3. 91 Detalles del cuestionario	113
Ilustración 3. 92 Crear nueva versión del cuestionario	114
Ilustración 3. 93 Vista de instrucciones del cuestionario	114
Ilustración 3. 94 Análisis macro	115
Ilustración 3. 95 Análisis macro (Puntajes por categorías)	115
Ilustración 3. 96 Análisis micro (Puntajes por variables de cada categoría)	116
Ilustración 3. 97 Análisis micro (Puntajes por variables de cada categoría)	116
Ilustración 3. 98 Análisis micro (Puntajes por pregunta de cada variable)	117
Ilustración 3. 99 Datos del alumno (Puntajes en general)	117
Ilustración 3. 100 Análisis de puntajes en general	118
Ilustración 3. 101 Grafico de puntajes	118

Ilustración 4. 1 Dashboard de los Alumnos	141
Ilustración 4. 2 Lista de Cuestionarios Pendientes	141
Ilustración 4. 3 Vista de Instrucciones del Cuestionario	141
Ilustración 4. 4 Vista del Cuestionario	142
Ilustración 4. 5 Vista del Cuestionario (Preguntas y Respuestas)	142
Ilustración 4. 6 Información importante del Alumno y Cuestionario (Resultados)	143
Ilustración 4. 7 Vista de la tabla de puntajes obtenidos	143
Ilustración 4. 8 Grafica araña de los puntajes obtenidos	144
Ilustración 4. 9 Dashboard del Docente	144
Ilustración 4. 10 Visualización de los Grupos del Docente	145
Ilustración 4. 11 Vista de la Información del Grupo y Cuestionarios Aplicados	145
Ilustración 4. 12 Tabla con la información de los alumnos del grupo seleccionado	146
Ilustración 4. 13 Tabla con la información de los alumnos del grupo seleccionado	146
Ilustración 4. 14 Tabla con la información de los alumnos del grupo seleccionado (Botones de acciones)	147
Ilustración 4. 15 Lista de los cuestionarios resueltos por alumno	147
Ilustración 4. 16 Vista de Puntajes Obtenidos (Información del alumno y cuestionario)	148
Ilustración 4. 17 Vista de Puntajes Obtenidos (Tabla de puntajes en las características)	148
Ilustración 4. 18 Vista de Puntajes Obtenidos (Grafica de araña)	149
Ilustración 4. 19 Vista de Puntajes Obtenidos en las características (Grafica de araña)	149
Ilustración 4. 20 Vista Macro de puntajes por categoría	150
Ilustración 4. 21 Vista Macro características que conforman el cuestionario	151
Ilustración 4. 22 Vista Macro características que conforman el cuestionario	152
Ilustración 4. 23 Vista Intermedio de las Expresiones	153
Ilustración 4. 24 Vista Intermedia de la gráfica de araña de las expresiones	154
Ilustración 4. 25 Vista Micro de las respuestas del alumno por pregunta	154

Ilustración 4. 26 Vista de la tabla de las Estadísticas Grupales	155
Ilustración 4. 27 Vista de la tabla de las Estadísticas Grupales	155
Ilustración 4. 28 Grafico de comparación de las Estadísticas Grupales	156
Ilustración 4. 29 Grafico de Comparación de las Estadísticas Grupales (Categorías y Expresiones)	156
Ilustración 4. 30 Información del Grupo de Primer Semestre (Sujetos de Prueba)	157
Ilustración 4. 31 Información del Grupo de Octavo Semestre (Sujetos de Prueba)	157
Ilustración 4. 32 Tabla con la información de los alumnos de prueba	158
Ilustración 4. 33 Tabla con la información de los alumnos de prueba	158
Ilustración 4. 34 Tabla con la información de los alumnos de prueba	159
Ilustración 4. 35 Vista de Puntajes Obtenidos por el alumno	159
Ilustración 4. 36 Vista de la Grafica araña de los puntajes obtenidos	160
Ilustración 4. 37 Vista de la Grafica araña de los puntajes obtenidos	160
Ilustración 4. 38 Resultado de la comparación de los dos grupos de prueba	161
Ilustración 4. 39 Resultado de la comparación de los dos grupos de prueba (Grafica araña)	161
Ilustración 4. 40 Resultado de la comparación de los dos grupos de prueba (Grafica araña)	162
Ilustración 4. 41 Vista de la información de los alumnos para la comparación.....	163
Ilustración 4. 42 Primera comparación entre alumnos (Primer y octavo semestre IIAS) .	163
Ilustración 4. 43Primera comparación entre alumnos grafica de araña (Primer y octavo semestre IIAS).....	164
Ilustración 4. 44 Segunda comparación entre alumnos (Primer y octavo semestre IIAS)	165
Ilustración 4. 45 Segunda comparación entre alumnos Grafica de araña (Primer y octavo semestre IIAS).....	166
Ilustración 4. 46 Tercera comparación entre alumnos (Primer y octavo semestre IIAS) .	167
Ilustración 4. 47 Tercera comparación entre alumnos Grafica de araña (Primer y octavo semestre IIAS).....	168
Ilustración 4. 48 Cuarta comparación entre alumnos (Primer y octavo semestre IIAS) ...	169
Ilustración 4. 49 Cuarta comparación entre alumnos Grafica de araña (Primer y octavo semestre IIAS).....	170
Ilustración 4. 50 Quinta comparación entre alumnos (Primer y octavo semestre IIAS) ...	171
Ilustración 4. 51 Quinta comparación entre alumnos Grafica de araña (Primer y octavo semestre IIAS).....	171
Ilustración 4. 52 Sexta comparación entre alumnos (Primer y octavo semestre IIAS)	172
Ilustración 4. 53 Sexta comparación entre alumnos Grafica de araña (Primer y octavo semestre IIAS).....	173

Anexo 1. Reunión de entrega del sistema.....	182
Anexo 2 Reunión de entrega del sistema con el equipo	182
Anexo 3 Evidencias de las pruebas de usabilidad al sujeto de prueba	183
Anexo 4 Evidencias de las pruebas de usabilidad al sujeto de prueba	183
Anexo 5 Evidencias de las pruebas de usabilidad al sujeto de prueba	184
Anexo 6 Evidencias de las pruebas de usabilidad al sujeto de prueba	184

Anexo 7. Migración tabla países	185
Anexo 8. Migración de la tabla Instituciones	186
Anexo 9. Migración de la tabla carreras	187
Anexo 10. Migración de la tabla generaciones	188
Anexo 11. Migración de la tabla grupos	189
Anexo 12. Migración de la tabla Cuestionarios	190
Anexo 13. Migración de la tabla Preguntas	191
Anexo 14. Migración de la tabla respuestas	192
Anexo 15. Migración de la tabla usuarios	193
Anexo 16. Migración de la tabla cuestionario grupo	194
Anexo 17. Migración de la tabla grupo con usuario	195
Anexo 18. Migración tabla categorías	196
Anexo 19. Migración tabla variables	197
Anexo 20. Migración tabla Pregunta Variable	198
Anexo 21. Migración tabla respuestas enviadas	199

Anexo 1. Migración tabla países	¡Error! Marcador no definido.
Anexo 2 Migración de la tabla Instituciones	¡Error! Marcador no definido.
Anexo 3 Migración de la tabla carreras	¡Error! Marcador no definido.
Anexo 4 Migración de la tabla generaciones	¡Error! Marcador no definido.
Anexo 5 Migración de la tabla grupos	¡Error! Marcador no definido.
Anexo 6 Migración de la tabla Cuestionarios	¡Error! Marcador no definido.
Anexo 7 Migración de la tabla Preguntas	¡Error! Marcador no definido.
Anexo 8 Migración de la tabla respuestas	¡Error! Marcador no definido.
Anexo 9 Migración de la tabla usuarios	¡Error! Marcador no definido.
Anexo 10 Migración de la tabla cuestionario grupo	¡Error! Marcador no definido.
Anexo 11 Migración de la tabla grupo con usuario	¡Error! Marcador no definido.
Anexo 12 Migración tabla categorías	¡Error! Marcador no definido.
Anexo 13 Migración tabla variables	¡Error! Marcador no definido.
Anexo 14 Migración tabla Pregunta Variable	¡Error! Marcador no definido.
Anexo 15 Migración tabla respuestas enviadas	¡Error! Marcador no definido.
Anexo 16 Migración de la tabla expresiones	200
Anexo 17 Migración de la tabla cuestionario resuelto	201
Anexo 18 Migración de la tabla cuestionario pregunta	202
Anexo 19 Migración de la tabla categorías expresiones	203
Anexo 20 Migración de la tabla puntajes por variables	204
Anexo 21 Migración de la tabla sumatorias de puntuaciones	206
Anexo 22 Modelo usuarios	207
Anexo 23 Relaciones del modelo usuarios	208
Anexo 24 Modelo países	209
Anexo 25 Modelo Instituciones	210
Anexo 26 Relaciones del modelo Instituciones	210
Anexo 27 Modelo Carreras	211
Anexo 28 Relaciones del modelo Carreras	¡Error! Marcador no definido.

Anexo 29 Modelo carrera grupos	212
Anexo 30 Modelo usuario grupo.....	213
Anexo 31 Modelo cuestionario.....	214
Anexo 32 Relaciones del modelo cuestionario.....	¡Error! Marcador no definido.
Anexo 33 Modelo cuestionario grupo	215
Anexo 34 Modelo cuestionario pregunta.....	216
Anexo 35 Modelo cuestionario resuelto	217
Anexo 36 Modelo expresiones	218
Anexo 37 Modelo generaciones.....	219
Anexo 38 Modelo grupos.....	220
Anexo 39 Relaciones del modelo grupos.....	¡Error! Marcador no definido.
Anexo 40 Modelo pregunta	221
Anexo 41 Modelo puntajes por variable	222
Anexo 42 Modelo respuesta enviada.....	223
Anexo 43 Modelo sumatoria puntajes	224
Anexo 44 Relaciones del modelo expresiones	¡Error! Marcador no definido.
Anexo 45 Modelo variable	225
Anexo 46 Validación de Datos en el Registro de Usuario	227
Anexo 47 Generación de Token y Respuesta del Registro	228
Anexo 48 Obtención de detalles de un usuario	229
Anexo 49 Respuesta de los detalles de usuario	230
Anexo 50 Actualización de datos de un usuario	231
Anexo 51 Obtención de los datos de todos los usuarios	232
Anexo 52 Respuesta de la obtención de datos	233
Anexo 53 Eliminación de un usuario	233
Anexo 54 Búsqueda de alumnos	234
Anexo 55 Respuesta de la búsqueda de alumnos	235
Anexo 56 Obtener maestros	235
Anexo 57 Respuesta de la obtención de información del maestro	236
Anexo 58 Obtención de datos de los administradores.....	237
Anexo 59 Respuesta de los datos del administrador	238
Anexo 60 Búsqueda de los datos del maestro por institución.....	239
Anexo 61 Respuesta de la búsqueda	240
Anexo 62 Búsqueda de alumnos por institución	241
Anexo 63 Respuesta de la búsqueda de alumnos	242
Anexo 64 Login de usuarios	243
Anexo 65 Controlador para crear países.....	244
Anexo 66 Actualización del país.....	245
Anexo 67 Obtención de todos los países.....	245
Anexo 68 Obtención de una institución específica	246
Anexo 69 Eliminación país	246
Anexo 70 Búsqueda de países	247
Anexo 71 Método para la creación de instituciones	248
Anexo 72 Método para la búsqueda de instituciones	249

Anexo 73 Método de búsqueda de instituciones por país.....	250
Anexo 74 Método para actualizar instituciones.....	250
Anexo 75 Método para eliminar instituciones.....	251
Anexo 76 Método para agregar y editar una carrera.....	251
Anexo 77 Métodos de búsqueda para las carreras	252
Anexo 78 Método para filtrar carreras por instituciones y eliminarlas	252
Anexo 79 Método para encontrar un cuestionario con sus preguntas	253
Anexo 80 Mapeo de las preguntas con sus respuestas.....	253
Anexo 81 Método para crear un cuestionario	254
Anexo 82 Asociación de las preguntas con un cuestionario	254
Anexo 83 Método para encontrar versiones de cuestionarios.....	255
Anexo 84 Método para publicar cuestionario	256
Anexo 85 Método para la obtención de cuestionarios por usuario	257
Anexo 86 Respuesta de la obtención del cuestionario	257
Anexo 87 Método para obtener la información de un solo cuestionario.....	258
Anexo 88 Método para duplicar y editar cuestionarios.....	258
Anexo 89 Método para marcar y obtener los cuestionarios resueltos.....	260
Anexo 90 Método para obtener los cuestionarios resueltos por alumno.....	260
Anexo 91 Método para obtener cuestionarios pendientes por alumno	261
Anexo 92 Método para obtener cuestionarios aplicados por grupo.....	261
Anexo 93 Método para agregar y editar una generación	262
Anexo 94 Método para obtener y eliminar una generación.....	263
Anexo 95 Método para buscar alumnos y obtener un grupo por el código de acceso ...	264
Anexo 96 Método para agregar un alumno a un grupo	264
Anexo 97 Método para crear grupos para el maestro.....	265
Anexo 98 Método para obtener grupos del maestro	265
Anexo 99 Método para obtener los grupos con su carrera e institución	266
Anexo 100 Método para obtener los integrantes del grupo	266
Anexo 101 Método para buscar a los alumnos por su nombre.....	267
Anexo 102 Método para agregar un alumno a un grupo	267
Anexo 103 Método para obtener todos los grupos	268
Anexo 104 Método para obtener las preguntas con sus variables	268
Anexo 105 Método para calcular los puntajes por variables	269
Anexo 106 Método para calcular el puntaje condicional	270
Anexo 107 Método para aplicar el puntaje condicional.....	270
Anexo 108 Método para guardar los puntajes a la BD.....	271
Anexo 109 Método para obtener los puntajes de un grupo	272
Anexo 110 Método para registrar las respuestas recibidas	272
Anexo 111 Método para guardar puntajes por categorías.....	273
Anexo 112 Método para obtener la sumatoria de puntajes por grupo	274
Anexo 113 Sumatoria de puntajes por categorías	274
Anexo 114 Obtención de la sumatoria de puntajes por expresiones	275
Anexo 115 Obtención de puntajes de preguntas por expresión.....	275
Anexo 116 Obtención de información de las preguntas, categorías y expresiones	276

Anexo 117 Respuesta de la obtención de los puntajes.....	276
Anexo 118 Obtención de puntajes por usuario y cuestionario	277
Anexo 119 Calculo de estadístico por grupo	278
Anexo 120 Calculo de cuartiles por grupo.....	279
Anexo 121 Calculo de la desviación estándar	279
Anexo 122 Calculo de cuartil.....	280
Anexo 123 Routers	281
Anexo 124 Routers	281
Anexo 125 Routers	282
Anexo 126 Routers	282
Anexo 127. Estructura del proyecto	283
Anexo 128. Estructura del proyecto (Frontend)	283
Anexo 129. Estructura del proyecto (Frontend)	284
Anexo 130. Librerías usadas del proyecto (Frontend).....	284
Anexo 131. Formulario del alumno	285
Anexo 132. Subscripciones para países e instituciones y rango de edad.....	286
Anexo 133. Funciones para obtener instituciones y carreras filtradas.....	287
Anexo 134. Función onSubmit	287
Anexo 135. Código para la vista del registro 1	288
Anexo 136. Código para la vista del registro 2	288
Anexo 137. Código para la vista del registro 3.....	289
Anexo 138. Código para la vista del registro 4	289
Anexo 139. Opciones del menú del header	290
Anexo 140. Contenedor principal	290
Anexo 141. Objetivo del cuestionario.....	291
Anexo 142. Código del Footer 1.....	291
Anexo 143. Código del Footer parte 2.....	292
Anexo 144. Código del Footer parte 3.....	292
Anexo 145. Código de contactos.....	293
Anexo 146. Código de "Quienes somos"	293
Anexo 147. Función de Login	293
Anexo 148. Código de la vista del login.....	294
Anexo 149.Código de la vista del menú.....	294
Anexo 150. Función para cargar información del alumno del menú.	295
Anexo 151. Código visto del dashboard.....	295
Anexo 152. Inicializador de funciones, funcion “formattedDate” y “getTotalGrupos”....	296
Anexo 153. Funciones “getTotalResultados” y “getTotalCuestionariosResueltos”	297
Anexo 154. Función “GetTotalCuestionariosPendientes”.....	297
Anexo 155. Funciones “buscarGrupo”, “unirseGrupo” y “cancelar”	298
Anexo 156. Código de la vista de la ventana buscar grupo	299
Anexo 157. Código de la vista de grupos.....	299
Anexo 158. Inicializador, Funciones “getAlumnos” y “enterGroup”.....	300
Anexo 159. Código de la vista de cuestionarios resueltos.	300
Anexo 160. Inicializador de cuestionarios resueltos.	301

Anexo 161. Código de la vista del apartado cuestionarios pendientes.	301
Anexo 162 . Inicializador y función “cargarGruposDelAlumno”.....	302
Anexo 163. Función “cargarCuestionariosPendientes”.....	302
Anexo 164. Función “irAcuestionario”.	303
Anexo 165. Inicializador del cuestionario y función “cargarCuestionario”.....	303
Anexo 166. Función “enviarRespuestas”.	304
Anexo 167. Funciones para la navegación del cuestionario.....	304
Anexo 168. Código de la vista del cuestionario parte 1.....	305
Anexo 169. Código de la vista del cuestionario parte 2.	305
Anexo 170. Código de la vista de resultados del cuestionario parte 1.....	305
Anexo 171. Código de la vista de resultados del cuestionario parte 2.....	306
Anexo 172. Código de la vista de resultados del cuestionario parte 3.....	306
Anexo 173. Inicializador de id de usuario y cuestionario.	307
Anexo 174. Obtener puntajes del cuestionario.....	307
Anexo 175. Función “createRadarChar”t parte 1.....	308
Anexo 176. Función “createRadarChar”t parte 2.....	309
Anexo 177. Código de la vista del menú del maestro.....	310
Anexo 178. Funciones para el menú del maestro.	310
Anexo 179. Función almacenar datos del grupo parte 1.....	311
Anexo 180. Función almacenar datos del grupo parte 1.....	311
Anexo 181. Inicializador, funciones “obtenerCuestionarios”, “obtenerGrupos” y “cargarPreguntas”.....	312
Anexo 182. Función “publicarCuestionario”.....	313
Anexo 183. Código la vista de información del cuestionario aplicado.....	313
Anexo 184. Inicializador y funciones “getGroupDetails”, “getAppliedQuizzes” y “getGroupMembers”.....	314
Anexo 185. Código de la vista del Dashboard del administrador.....	315
Anexo 186. Código de la vista del menú del administrador.....	315
Anexo 187. Funciones “loadCountries” y “filteredCountries”.....	316
Anexo 188. Código componente agregar país.....	316
Anexo 189. Funcionalidad de la vista lista de instituciones.	317
Anexo 190. Función de Instituciones filtradas por país.....	317
Anexo 191. Funcionalidad de la lista de maestros por institución.	318
Anexo 192. Código del formulario del maestro.	318
Anexo 193. Funcionalidad para obtener los datos del maestro y los miembros del grupo.	319
Anexo 194. Funcionalidad de lista de alumnos de la institución.....	320
Anexo 195. Funcionalidad de lista de alumnos de la institución.....	320
Anexo 196. Funcionalidad de lista de maestros de la institución.	321
Anexo 197. Funcionalidades de la lista de cuestionarios.....	322
Anexo 198. Funcionalidades para la vista detalles del cuestionario.....	323
Anexo 199. Función “cargarCuestionarioOriginal”.....	323
Anexo 200. Función “crearCuestionario”.....	324
Anexo 201. Código de la vista del formulario para crear nueva versión del cuestionario.	325

Anexo 202. Modelo Responses	326
Anexo 203. Modelo usuarios parte 1	326
Anexo 204. Modelo usuarios parte 2.	327
Anexo 205. Modelo usuarios parte 3.	328
Anexo 206. Modelo catálogos parte 1.	329
Anexo 207. Modelo catálogos parte 2.	329
Anexo 208. Modelo Cuestionarios.	330
Anexo 209. Modelo grupos.....	330
Anexo 210. Servicio Autenticación.....	331
Anexo 211. “userService” parte 1.	332
Anexo 212. “userService” parte 2.	332
Anexo 213. “userService” parte 3.	333
Anexo 214. “catalogsService” parte 1.....	333
Anexo 215. “catalogsService” parte 2.....	333
Anexo 216. “AlumnosGruposService”.....	334
Anexo 217. “cuestionarioService” parte 1.....	334
Anexo 218. “cuestionarioService” parte 2.....	334
Anexo 219. “cuestionarioService” parte 3.....	335
Anexo 220. “PuntajeService”.....	335
Anexo 221. “PuntajeService”.....	335
Anexo 222. “SolvedQuizzesService”.....	336
Anexo 223. “TeacherGroupService” parte 1.	336
Anexo 224. “TeacherGroupService” parte 2.	337
Anexo 225. Validador de confirmación de contraseña.....	337
Anexo 226. Email validado	338
Anexo 227. Validador de longitud	338
Anexo 228. Validador de campo requerido.....	339
Anexo 229. Validador de nombre.....	339
Anexo 230. Validadora contraseña confirmada.....	340
Anexo 231. Validador de verificación de matrícula o correo	340
Anexo 232. Routers página principal	341
Anexo 233. Routers página principal (Parte 2)	341
Anexo 234 Routers página principal (Parte 3).....	342
Anexo 235. Routes para el dashboard de alumnos (parte 1)	342
Anexo 236 Routes para el dashboard de alumnos (Parte 2)	343
Anexo 237 Routes para el dashboard de maestro (Parte 1)	343
Anexo 238 Routes para el dashboard de maestro (Parte 2)	344
Anexo 239. Routes para el dashboard de maestro (Parte 3).....	344
Anexo 240. Routes para el dashboard de administrador (Parte 1).....	345
Anexo 241. Routes para el dashboard de administrador (Parte 2).....	345
Anexo 242. Routes para el dashboard de administrador (Parte 3).....	346
Anexo 243. Routes para el dashboard de administrador (Parte 4).....	346
Anexo 244. Routes para el dashboard de administrador (Parte 5).....	347

Índice Tablas.

Tabla 3. 1 Equipo Scrum	51
Tabla 3. 2 Tipos de usuarios del sistema	52
Tabla 3. 3 Requerimientos funcionales	55
Tabla 3. 4 Requerimientos no funcionales	56
Tabla 3. 5 Tabla de pruebas exploratorias	122
Tabla 3. 6 Caso de Prueba de Usabilidad Menú Principal.....	123
Tabla 3. 7 Caso de Prueba de Usabilidad Inicio de Sesión.....	123
Tabla 3. 8 Caso de Prueba de Usabilidad Perfil del Alumno	124
Tabla 3. 9 Caso de Prueba de Usabilidad Cuestionarios Pendientes	124
Tabla 3. 10 Caso de Prueba de Usabilidad Respuesta a cuestionarios	125
Tabla 3. 11 Caso de Prueba de Usabilidad Retroalimentación Personalizada	125
Tabla 3. 12 Caso de Prueba de Usabilidad Progreso Individual	126
Tabla 3. 13 Caso de Prueba de Usabilidad Perfil de Docentes	126
Tabla 3. 14 Caso de Prueba de Usabilidad Vistas de Resultados.....	127
Tabla 3. 15 Caso de Prueba de Usabilidad Exportar Resultados de un Cuestionario.....	128
Tabla 3. 16 Caso de Prueba de Usabilidad Creacion y Gestión de Grupos	129
Tabla 3. 17 Caso de Prueba de Usabilidad Estadísticas del Grupo	129
Tabla 3. 18 Caso de Prueba de Usabilidad Comparación Entre Grupos.....	130
Tabla 3. 19 Caso de Prueba de Usabilidad Generación de Informes	130
Tabla 3. 20 Caso de Prueba de Usabilidad Gestión de Usuarios	131
Tabla 3. 21 Caso de Prueba de Usabilidad Resultados de Cuestionarios.....	131
Tabla 3. 22 Caso de Prueba de Usabilidad Generación de Informes Detallados	132
Tabla 3. 23 Caso de Prueba de Usabilidad Consulta de Datos de Perfil de Usuarios	133
Tabla 3. 24 Casos de Pruebas de Aceptación	136
Tabla 4. 1 Comparación Entre Perfil de Egreso y Características Emprendedoras	139

Resumen.

El presente proyecto tiene como objetivo desarrollar un sistema de inteligencia de negocios destinado a identificar patrones en las competencias emprendedoras de egresados universitarios. El sistema analizara el desarrollo de los estudiantes durante su carrera, comparando sus competencias con el perfil de egreso definido en su plan de estudios. De esta manera, se podrá evaluar si los estudiantes han alcanzado el perfil emprendedor esperado al finalizar su formación.

El desarrollo del sistema se basa en una estructura modular que incluye la creación de dashboards específicos para alumnos, maestros y administradores, así como la implementación de una base de datos y API REST que gestionen la interacción del sistema. Además, se integrará un componente para la comparación de grupos, usando estadísticas y gráficos visuales que permitan monitorear el progreso de los estudiantes a lo largo de su carrera. El proyecto también abarca la creación de un sistema de autenticación para los usuarios.

Los resultados obtenidos hasta el momento sugieren que el sistema será útil para identificar y seguir el desarrollo de los estudiantes, facilitando la toma de decisiones para mejorar la formación de futuros emprendedores y optimizando la gestión del talento en las universidades, incrementando las probabilidades de éxito en el ámbito empresarial. En términos de impacto, el proyecto ofrece una herramienta innovadora que optimiza la gestión del talento emprendedor en universidades, mejorando así la empleabilidad de sus egresados.

Capítulo 1

1.1 INTRODUCCIÓN

El presente proyecto tiene como objetivo desarrollar un sistema de inteligencia de negocios que facilite la identificación de patrones en las competencias emprendedoras de los egresados universitarios en un estudio longitudinal, a partir de su perfil de egreso. Este sistema surge como respuesta a la necesidad de contar con herramientas que permitan a las instituciones educativas detectar el potencial emprendedor en sus estudiantes, brindándoles apoyo personalizado para fortalecer sus competencias y aumentar sus probabilidades de éxito en el ámbito empresarial.

El emprendimiento es reconocido como un factor clave para el crecimiento económico y social, especialmente en entornos donde las oportunidades laborales tradicionales son limitadas. En este sentido, los egresados universitarios representan un grupo con un alto potencial emprendedor debido a su formación académica y la variedad de conocimientos adquiridos. Sin embargo, muchos carecen de las herramientas necesarias para identificar sus habilidades emprendedoras, lo que limita su capacidad de establecer negocios sostenibles. Este proyecto busca contribuir a resolver esta problemática proporcionando una herramienta que permita a las universidades realizar un seguimiento del desarrollo de sus estudiantes y evaluar si cumplen con los estándares de perfil de egreso esperados.

El sistema propuesto utilizara técnicas de inteligencia de negocios para analizar la evolución de las competencias emprendedoras en los egresados, generando información relevante para la toma de decisiones en las instituciones educativas. A través de dashboards personalizados y la comparación de estadísticas de grupos, el sistema ayudara a las universidades a evaluar el progreso de los estudiantes y la efectividad de los planes de estudio en el desarrollo de estas competencias.

El alcance del proyecto incluye el diseño y desarrollo de dashboards personalizados para alumnos, maestros y administradores, así como la creación de una base de datos estructurada que permitirá la visualización de resultados y el análisis de datos. Además, se implementará un sistema de autenticación para el registro de usuarios, junto con la creación de API REST para gestionar la interacción entre los componentes del sistema.

El proyecto se desarrollará en fases, comenzando con el análisis de requerimientos y el diseño conceptual del sistema, seguido de la implementación de módulos básicos y avanzados, hasta la entrega final y puesta en marcha del sistema en un entorno de producción. Las limitaciones del proyecto incluyen la exclusión de niveles educativos que no sean universitarios y la falta de seguimiento una vez que el egresado inicie su emprendimiento.

1.2 ANTECEDENTES. Y/O MARCO TEÓRICO

El emprendimiento ha emergido como un elemento clave en el desarrollo económico y social, especialmente en contextos donde las oportunidades laborales son limitadas. Según (Laat, 2023), el emprendimiento no solo impulsa la economía, sino que también promueve la innovación y el desarrollo sostenible, lo que lo convierte en un aspecto crucial en la educación superior. La capacidad de emprender es esencial para los egresados universitarios, quienes, a pesar de poseer una sólida formación académica, a menudo carecen de las herramientas necesarias para identificar y desarrollar sus habilidades emprendedoras.

Como lo menciona (Gutiérrez Montoya, 2011) las instituciones educativas tienen un papel crucial en el fomento de la mentalidad emprendedora entre los estudiantes, integrando currículos que promuevan la innovación y el desarrollo de competencias clave. Estas instituciones no solo deben enfocarse en impartir conocimientos académicos, sino también en proveer herramientas que permitan a los estudiantes identificar y desarrollar competencias que fortalezcan su capacidad de emprendimiento.

La implementación de programas que fomenten el emprendimiento en las instituciones educativas se ha vuelto indispensable. Las universidades tienen un papel fundamental en la formación de competencias emprendedoras, ya que pueden integrar en su currículo programas que permitan a los estudiantes identificar su potencial y desarrollar un enfoque empresarial (Laat, 2023). Sin embargo, muchas instituciones todavía enfrentan desafíos para implementar programas efectivos que identifiquen y promuevan estas habilidades en sus egresados. En particular, es necesario contar con metodologías eficientes que permitan no solo evaluar, sino también mejorar las competencias emprendedoras.

De acuerdo con (Lizbeth Alicia González Tamayo, 2019), la evaluación de las competencias emprendedoras a través de instrumentos estandarizados permite a las instituciones educativas identificar perfiles emprendedores y proponer estrategias de mejora personalizadas para los estudiantes. Existen diversas metodologías para evaluar las competencias emprendedoras en los estudiantes. Según (Gloria Y. Rojas, 2018), la identificación de las características personales emprendedoras es clave en la formación de emprendedores en el ámbito educativo. Los docentes utilizan estrategias pedagógicas diseñadas específicamente para fomentar estas competencias, lo que subraya la importancia de instrumentos estandarizados en la evaluación de dichas características, entre ellos, el uso de cuestionarios estandarizados que permiten identificar las Características Emprendedoras Personales (CEPs). Estos instrumentos son útiles para proporcionar a los estudiantes una retroalimentación clara sobre sus capacidades emprendedoras y guiarles hacia el desarrollo de planes de acción concretos. Además, la aplicación de escalas como la de Likert en estos cuestionarios proporciona una medición precisa del grado de acuerdo o desacuerdo de los estudiantes con afirmaciones relacionadas con las competencias emprendedoras, lo que facilita la identificación de áreas de mejora.

La escala de Likert es ampliamente utilizada en este tipo de cuestionarios para medir el grado de acuerdo o desacuerdo con afirmaciones relacionadas con las competencias emprendedoras. Este enfoque no solo permite una medición precisa de las características individuales, sino que también facilita la comparación de resultados y la identificación de áreas de mejora. La retroalimentación

generada a partir de esta evaluación permite que las instituciones educativas proporcionen recomendaciones más personalizadas y orientadas a fortalecer las capacidades emprendedoras en sus egresados.

Además, la escala de Likert se ha convertido en una herramienta esencial en la medición de competencias emprendedoras. Según (Bloom, 2024) , esta escala facilita la cuantificación del acuerdo o desacuerdo con afirmaciones específicas, lo que permite una evaluación precisa de las características individuales de los estudiantes y proporciona una retroalimentación más detallada sobre las áreas de mejora.

No obstante, la mayoría de estos sistemas carecen de un análisis integral que permita a las instituciones educativas ofrecer recomendaciones personalizadas basadas en los resultados obtenidos. Además, no se han explorado en profundidad las posibilidades de utilizar tecnologías emergentes, para identificar patrones a largo plazo en el desarrollo de las competencias emprendedoras. Contar con un sistema que analice estos patrones facilitaría la generación de estrategias educativas mas efectivas y la mejora continua en la formación de emprendedores.

Según (Mariana, 2024), las autoevaluaciones emprendedoras son un recurso valioso para guiar a los estudiantes hacia el reconocimiento de sus fortalezas y debilidades emprendedoras, facilitando el diseño de planes de acción personalizados. Sin embargo, el uso de herramientas más avanzadas podría mejorar este proceso al identificar tendencias y patrones en los resultados de los cuestionarios, proporcionando así un enfoque más estructurado en la formación de competencias emprendedoras.

Por lo cual, dada esta situación, el presente proyecto propone el desarrollo de un sistema de inteligencia de negocios que permita a las instituciones educativas identificar patrones en las competencias emprendedoras de sus egresados en un estudio longitudinal. A través de la aplicación de un cuestionario estandarizado para evaluar las CEPs, el sistema proporcionará puntajes que reflejen las capacidades emprendedoras de los usuarios, junto con recomendaciones para su mejora. Este enfoque busca llenar un vacío existente en el ámbito educativo y emprendedor, contribuyendo a la formación de un ecosistema emprendedor más sólido y sostenible.

1.2.1 PLANTEAMIENTO

El Instituto Tecnológico de Calkiní, Campeche (ITESCAM), junto con otros tecnológicos mexicanos, enfrenta el reto de promover el emprendimiento como una competencia clave en sus egresados. Aunque el Tecnológico Nacional de México (TecNM) ha implementado un marco regulatorio que establece una base curricular común para la promoción del emprendedurismo, el proceso de identificación, evaluación y mejora de los perfiles emprendedores de los estudiantes sigue siendo un desafío considerable.

La implementación de programas de emprendedurismo se enfrenta a varios obstáculos en las instituciones educativas de nivel superior, entre los que destacan la falta de herramientas adecuadas para la identificación temprana de las características emprendedoras personales (CEPs) en los estudiantes y la ausencia de sistemas que permitan el seguimiento y evaluación eficaz de las competencias adquiridas a lo largo del programa.

A nivel institucional, el Modelo de Incubación de Empresas del TecNM proporciona lineamientos generales para el fomento del emprendimiento, pero las instituciones deben adaptar y mejorar este programa según sus necesidades específicas. Actualmente, ITESCAM carece de una plataforma tecnológica que permita evaluar de manera sistemática y continua las habilidades emprendedoras de los estudiantes, lo que dificulta la personalización de los planes formativos y el seguimiento individualizado del progreso de los alumnos.

Además, los datos obtenidos de las evaluaciones de los estudiantes no se están utilizando de manera eficiente para generar análisis que permitan la toma de decisiones informadas. Esta falta de integración de un sistema de inteligencia de negocios impide que las autoridades educativas puedan visualizar tendencias, identificar áreas de mejora, o generar informes detallados sobre los avances en los perfiles emprendedores de los estudiantes.

Dado esta problemática, surge la necesidad de desarrollar un sistema de inteligencia de negocios que permita evaluar las Características Emprendedoras Personales (CEPs) de los estudiantes, generar informes detallados y proporcionar una plataforma para la gestión estratégica de estos perfiles. En este sentido, resulta clave responder a la siguiente pregunta de investigación:

¿Cómo puede un sistema de inteligencia de negocios identificar patrones en las competencias emprendedoras de los egresados universitarios en un estudio longitudinal, a partir de su perfil de egreso?

Este sistema no solo facilitaría la identificación de las competencias emprendedoras de los egresados, sino que también serviría como herramienta para que las instituciones educativas gestionen de manera más eficiente su programa de emprendedurismo, alineándose con los objetivos del proyecto “Transformando la Educación Superior”, financiado por la Fundación W.K. Kellogg.

La implementación de este sistema permitirá al ITESCAM y otras instituciones similares mejorar la gestión de su programa emprendedor, al proporcionar datos claros y procesables sobre el rendimiento de los estudiantes, facilitando el diseño de actividades educativas más efectivas, y

garantizando un seguimiento más robusto de los resultados.

1.2.2 JUSTIFICACIÓN

El emprendimiento es un motor esencial para el desarrollo económico y social, especialmente en contextos de oportunidades laborales limitadas. Los egresados universitarios, por su nivel educativo y diversidad de conocimientos, representan un grupo con alto potencial emprendedor; sin embargo, muchos carecen de las herramientas necesarias para desarrollar estas habilidades, lo que limita su éxito en la creación de negocios sostenibles. Este proyecto busca resolver este problema mediante un sistema de inteligencia de negocios que permita identificar y mejorar los perfiles emprendedores de los egresados. La aplicación de este sistema no solo identificará a aquellos con alto potencial, sino que también ofrecerá recomendaciones personalizadas para fortalecer sus competencias, incrementando así sus posibilidades de éxito en el ámbito empresarial.

La realización de este proyecto es esencial porque aborda una necesidad actual en el ámbito educativo y empresarial: la identificación y formación de futuros emprendedores. Al utilizar técnicas avanzadas de inteligencia de negocios, se pretende transformar la manera en que las instituciones educativas y los egresados abordan el desarrollo de habilidades emprendedoras. Esto beneficiará directamente a los egresados al proporcionarles un camino claro hacia el emprendimiento, y, de manera indirecta, a las universidades, al mejorar la tasa de éxito de sus egresados en el mercado laboral.

El aporte de este proyecto será significativo tanto a nivel académico como práctico. Académicamente, contribuirá al campo de la inteligencia de negocios aplicada al desarrollo emprendedor, mientras que, en la práctica, ofrecerá una herramienta innovadora que puede ser adoptada por universidades y otras instituciones para mejorar sus programas de apoyo al emprendimiento. No obstante, este proyecto también presenta ciertas limitaciones, como la necesidad de acceso a datos de calidad y la correcta interpretación de estos, así como la implementación de las recomendaciones generadas por el sistema.

1.3 OBJETIVOS GENERALES Y ESPECIFICOS.

1) Objetivo General

Desarrollar un sistema de inteligencia de negocios que permita la identificación y mejora de perfiles emprendedores entre los egresados universitarios, con el fin de potenciar sus capacidades y facilitar su inserción en el ecosistema emprendedor.

2) Objetivos específicos

- Identificar las características clave que definen a un perfil emprendedor exitoso entre los egresados universitarios, utilizando análisis de datos y técnicas de inteligencia de negocios.
- Diseñar e implementar un sistema de inteligencia de negocios que permita clasificar y evaluar los perfiles emprendedores, basado en los datos recopilados de egresados universitarios.
- Proporcionar recomendaciones personalizadas para la mejora de los perfiles emprendedores identificados, utilizando los resultados del análisis de datos.

Evaluar la efectividad del sistema en la identificación y mejora de perfiles emprendedores, mediante pruebas piloto en una universidad seleccionada.

1.4 ALCANCES Y LIMITACIONES.

Alcance del proyecto.

El sistema de inteligencia de negocios destinado a identificar y mejorar los perfiles emprendedores entre los egresados universitarios tendrá las siguientes características:

- ❖ Interfaz simple, dinámica e intuitiva para un fácil manejo
- ❖ Creación de dashboards personalizados para distintos roles (alumnos, maestros, administradores).
- ❖ Implementación de un sistema de autenticación y registro de usuarios para el correcto ingreso al sistema.
- ❖ Integración de API REST para gestionar y visualizar datos.
- ❖ Permitirá a los usuarios la descarga de su informe de resultados en formatos PDF
- ❖ Todos los datos ingresados por los usuarios, así como sus respuestas serán guardadas en una base de datos
- ❖ Realización de pruebas iniciales y corrección de errores.
- ❖ Capacitación para los usuarios finales del sistema.
- ❖ Documentación completa del sistema.
- ❖ Despliegue final del sistema en un servidor.

Limitaciones.

- La plataforma solo será desarrollada como un sitio web
- Los usuarios son registrados con la matrícula proporcionada por las instituciones
- Se necesitará el acceso a internet para utilizar el sistema, lo que puede ser un desafío en áreas con limitaciones de conectividad
- El sistema se enfocará exclusivamente en perfiles emprendedores de egresados universitarios.
- No incluirá el seguimiento o evaluación de emprendimientos una vez iniciados.
- No abordará el desarrollo de habilidades emprendedoras en sectores no educativos.
- El proyecto se limitará al contexto educativo universitario, sin incluir otros niveles académicos.

Capítulo 2. MARCO TEORICO

Este marco teórico proporciona los fundamentos conceptuales necesarios para el desarrollo del sistema de inteligencia de negocios enfocado en la identificación y mejora de perfiles emprendedores. En este capítulo, se explorarán los conceptos clave relacionados con el emprendimiento, la inteligencia de negocios, las características emprendedoras personales (CEPs) y la metodología Scrum, que guiarán la ejecución del proyecto.

2.1. EMPRENDIMIENTO

El emprendimiento es un concepto clave en el desarrollo económico y social, ya que implica la capacidad de identificar oportunidades, asumir riesgos calculados y gestionar recursos para generar valor. A continuación, se desglosan los principales aspectos del emprendimiento para comprender su alcance y relevancia en diversos contextos.

2.1.1. Definición de Emprendimiento

El emprendimiento se define como el proceso mediante el cual un individuo o grupo identifica una oportunidad, reúne recursos y los utiliza para crear una nueva empresa o mejorar una existente (Que es el emprendimiento, 2023). Este proceso se caracteriza por la innovación, la toma de decisiones estratégicas y la gestión de riesgos.

Según (M.A. Alberto Bucardo Castro, 2015), el emprendedor es un agente de cambio que introduce nuevas combinaciones de productos, servicios o procesos, impulsando el crecimiento económico. En un contexto más contemporáneo, el emprendimiento no solo implica la creación de empresas, sino también la capacidad de resolver problemas sociales mediante iniciativas innovadoras.

2.1.2. Tipos de Emprendimiento.

El emprendimiento puede clasificarse en diversos tipos según su propósito, ámbito y modelo de negocio. Entre los más destacados se encuentran:

- **Emprendimiento tradicional:** Se enfoca en la creación de negocios con fines de lucro. Ejemplo: una tienda de comercio minorista.
- **Emprendimiento social:** Su objetivo principal resolver problemas sociales o ambientales. Ejemplo: organizaciones que promueven el acceso al agua potable.
- **Emprendimiento tecnológico:** Basado en la innovación tecnológica para desarrollar nuevos productos o servicios. Ejemplo: startups de inteligencia artificial.
- **Emprendimiento corporativo:** También conocido como intraemprendimiento, ocurre dentro de empresas establecidas y busca desarrollar nuevas líneas de negocio o innovar en procesos internos.

Cada tipo de emprendimiento responde a diferentes necesidades del mercado, lo que resalta la importancia de adaptarse a las demandas y contextos específicos.

2.1.3. Fases del Proceso Emprendedor

El proceso emprendedor es dinámico y se desarrolla en varias etapas interrelacionadas (Ventana | EBC, 2024):

- **Identificación de oportunidades:** Los emprendedores detectan problemas o necesidades que pueden transformarse en soluciones innovadoras.
- **Planificación:** Incluye el diseño del modelo de negocio, la elaboración de un plan financiero y la definición de estrategias de mercado.
- **Desarrollo y lanzamiento:** En esta etapa se pone en marcha el proyecto, incluyendo la producción, distribución y comercialización del producto o servicio.
- **Crecimiento y consolidación:** Implica la expansión del negocio, la mejora continua de procesos y la fidelización de clientes.
- **Madurez o transformación:** El negocio alcanza su estabilidad o redefine su rumbo para adaptarse a nuevos desafíos del mercado.

Estas fases no siempre se presentan de manera lineal, ya que el entorno emprendedor es altamente volátil y exige flexibilidad en la toma de decisiones.

2.1.4. Retos y Oportunidades en el Emprendimiento

En México, los emprendedores enfrentan retos significativos como la falta de financiamiento, burocracia y limitaciones tecnológicas. Sin embargo, también cuentan con oportunidades derivadas de la creciente digitalización, el acceso a mercados internacionales y programas de apoyo gubernamental. La innovación y la capacidad de adaptarse a entornos dinámicos son esenciales para superar obstáculos y aprovechar nuevas oportunidades (Carbajal, México Emprende, 2023).

El emprendimiento enfrenta múltiples desafíos, pero también ofrece innumerables oportunidades:

Retos:

- **Acceso a financiamiento:** Muchos emprendedores carecen de recursos financieros suficientes para iniciar o escalar sus proyectos.
- **Competencia en mercados saturados:** La globalización ha incrementado la rivalidad entre negocios, especialmente en sectores tradicionales.
- **Adaptación tecnológica:** Las empresas deben mantenerse actualizadas en un entorno en constante evolución tecnológica.
- **Marco regulatorio:** En algunos países, las regulaciones excesivas dificultan la creación y operación de nuevos negocios.

Oportunidades:

- **Mercados emergentes:** La creciente demanda de soluciones innovadoras en sectores como la sostenibilidad o la salud representa una oportunidad significativa.
- **Transformación digital:** El acceso a herramientas tecnológicas facilita la creación y gestión de empresas con costos más bajos y mayor alcance.
- **Conexión global:** Internet permite a los emprendedores acceder a nuevos mercados y establecer redes de colaboración internacionales.
- **Impacto social:** El emprendimiento social está ganando reconocimiento como un medio efectivo para abordar problemas globales, como la desigualdad y el cambio climático.

2.2. INTELIGENCIA DE NEGOCIOS

2.2.1. Definición de Inteligencia de Negocios

La inteligencia de negocios (BI, por sus siglas en inglés) es un conjunto de estrategias, procesos, tecnologías y herramientas que permite a las organizaciones transformar datos en información útil para la toma de decisiones. Este concepto está centrado en la gestión del conocimiento, ya que busca aprovechar los datos para identificar patrones, tendencias y oportunidades, contribuyendo al desarrollo de ventajas competitivas. En México, esta metodología ha ganado relevancia por su capacidad de mejorar la eficiencia operativa y el rendimiento estratégico de las empresas, especialmente en sectores tecnológicos y administrativos (Carbajal, México Emprende, 2026).

2.2.2. Componentes de la Inteligencia de Negocios

Los componentes principales de la inteligencia de negocios incluyen (Eduardo Ahumada Tello, 2015):

- **Sistemas de Información:** Son herramientas que gestionan la entrada, almacenamiento y salida de datos. Entre ellas, destacan los sistemas CRM (Gestión de Relaciones con Clientes) y ERP (Planificación de Recursos Empresariales), ampliamente utilizados en México para integrar procesos y mejorar la productividad.
- **Proceso de Análisis:** Este componente implica el uso de técnicas estadísticas, minería de datos y algoritmos de aprendizaje automático para encontrar patrones y correlaciones significativas. En México, estos métodos son esenciales para empresas de sectores como manufactura y servicios financieros.
- **Visualización de Datos:** Se refiere al uso de gráficos interactivos, tableros digitales y mapas de calor para presentar datos de manera comprensible y útil. Herramientas como Power BI y Tableau son populares en empresas mexicanas debido a su capacidad para simplificar datos complejos.
- **Gestión del Conocimiento:** La capacidad de convertir datos en acciones efectivas es el núcleo de la inteligencia de negocios. Este proceso incluye la creación, disseminación y aplicación de conocimiento para mejorar los procesos empresariales.

2.2.3. Herramientas de Inteligencia de Negocios

En México, se han adoptado múltiples herramientas de BI que abarcan desde plataformas comerciales hasta soluciones personalizadas (Eduardo Ahumada Tello, 2015):

- **Power BI:** Amplia aceptación por su integración con productos de Microsoft y su facilidad de uso.
- **Tableau:** Conocida por sus capacidades avanzadas de visualización de datos.
- **QlikView:** Popular entre empresas medianas por su enfoque en análisis interactivo (Poyatos, 2024).

Además, plataformas como Google Data Studio y sistemas de código abierto como Pentaho son utilizadas por emprendedores para minimizar costos y maximizar resultados. Estas herramientas

ayudan a interpretar grandes volúmenes de datos, optimizando procesos como logística, ventas y atención al cliente

2.2.4. Aplicaciones de la Inteligencia de Negocios en el Emprendimiento

La inteligencia de negocios tiene aplicaciones prácticas en el ámbito emprendedor, ofreciendo beneficios como (Tableau, 2024):

- **Análisis del Mercado:** Permite a los emprendedores en México identificar tendencias, evaluar la competencia y segmentar mercados, lo que es esencial para lanzar productos o servicios efectivos.
- **Optimización de Recursos:** Mediante análisis predictivos, los emprendedores pueden asignar recursos de manera eficiente, reduciendo costos y mejorando la rentabilidad. Este enfoque es particularmente relevante para pequeñas y medianas empresas mexicanas.
- **Monitoreo del Desempeño:** Los emprendedores pueden usar tableros de BI para rastrear métricas clave en tiempo real, como ventas, inventarios y satisfacción del cliente, lo que les permite reaccionar rápidamente ante cambios en el mercado.
- **Innovación:** La inteligencia de negocios fomenta la creación de nuevos modelos de negocio y estrategias de mercado basadas en datos. En México, esto ha llevado al desarrollo de soluciones tecnológicas y sociales innovadoras que abordan problemas locales.

2.3. LAS CARACTERÍSTICAS EMPRENDEDORAS PERSONALES (CEPS)

2.3.1. Definición de CEPs

Las Características Emprendedoras Personales (CEPs) son un conjunto de habilidades, actitudes, y comportamientos que distinguen a los emprendedores exitosos de otras personas. Estas características se relacionan con la capacidad de identificar oportunidades, gestionar recursos, asumir riesgos y liderar iniciativas. En México, las CEPs son vistas como un elemento crucial en el desarrollo del ecosistema emprendedor, especialmente en un contexto donde las pequeñas y medianas empresas (PyMEs) representan un alto porcentaje de la economía nacional. El desarrollo de las CEPs se sustenta en teorías psicológicas y de comportamiento organizacional que identifican rasgos como la autoconfianza, la resiliencia, y la capacidad de innovar como esenciales para el emprendimiento efectivo (Chisparural, 2022).

2.3.2. Principales Características Emprendedoras

Entre las CEPs (J., 2018) mas relevantes identificadas en la literatura, se encuentran las siguientes:

a) ATRIBUTO EMPRENDEDOR

- **Buscar y aprovechar las oportunidades:** Los emprendedores mexicanos destacan por su capacidad para identificar necesidades insatisfechas en el mercado y desarrollar

soluciones innovadoras. Esta habilidad les permite crear propuestas de valor únicas y competitivas. (Blog, 2023)

- **Fijar objetivos:** *Establecer metas claras y alcanzables es fundamental para orientar los esfuerzos y recursos de manera efectiva. Los emprendedores exitosos en México suelen definir objetivos específicos que guían el crecimiento de sus negocios. (Empleo, s.f.)*
- **Planificar y hacer seguimiento sistemático:** *La planificación estratégica y el monitoreo constante del progreso permiten a los emprendedores adaptarse a cambios y optimizar sus operaciones. Esta práctica es común entre los emprendedores mexicanos que buscan sostenibilidad a largo plazo (Laura Iturbide Galindo, 2020)*

b) ACTITUD EMPRENDEDORA

- **Ser persistentes:** *Como emprendedores, tener persistencia es fundamental para poder alcanzar el éxito, porque nos permite levantarnos y seguir adelante a pesar de los obstáculos. Esta actitud nos permite superar las adversidades y mantener el enfoque en nuestros objetivos. (Blanco, 2021)*
- **Cumplimiento de compromisos:** *Es una cualidad importante y virtuosa, ya que puede asegurar el éxito en proyectos futuros. Los emprendedores deben de honrar recuerdos con sus clientes, proveedores y socios esto es esencial para construir una reputación sólida y relaciones de confianza. (Cavero, Cumplimiento de compromisos, 2023)*
- **Exigir eficiencia y calidad:** *En un emprendimiento significa que debemos de buscar hacer las cosas de manera más rápida, mejor y más económica. Esto implica cumplir las normas de calidad y excelencia. Como emprendedores debemos de buscar constantemente mejorar la calidad de nuestros productos o servicios y optimizar procesos para satisfacer las expectativas del mercado. (Indeed, 2024)*

c) CAPACIDAD EMPRENDEDORA

- **Correr riesgos moderados:** *Esta cualidad implica que el emprendedor debe de evaluar cuidadosamente las posibles consecuencias antes de tomar decisiones que puedan ocasionar un riesgo. No se trata de evitar el riesgo por completo, pero si debemos de asumir aquellos que son calculados y que nos ofrezcan una relación favorable entre el riesgo y beneficio. (Pedraza, 2023)*
- **Tener autoconfianza:** *Esta característica se refiere a la seguridad que el emprendedor debe de sentir en sus propias habilidades y juicios. Esta confianza y seguridad que sienten les permite enfrentar desafíos y tomar las decisiones con determinación, incluso en las situaciones de incertidumbre. (Pedraza, 2023)*

d) CAPACIDAD INTERPERSONAL

- **Buscar y conseguir información:** Esta habilidad consiste en la proactividad del emprendedor para obtener información relevante que les ayude en la toma de decisiones. Los emprendedores necesitamos estar al tanto de las últimas tendencias, oportunidades del mercado y las necesidades que tiene sus clientes para poder crear sus productos o servicios que satisfagan estas necesidades y que al mismo tiempo sean exitosas en el mercado. (Cavero, Búsqueda de información. 3ª Competencia emprendedora, 2023)
- **Persuasión y creación de redes de apoyo:** Esta capacidad en un emprendedor significa influir en otras personas y establecer una red de contactos que le puedan brindar apoyo en los diferentes aspectos de su emprendimiento. Es decir que deben de tener la habilidad para negociar, convencer y mantener relaciones beneficiosas con los clientes, proveedores e inversores. (Bernal, 2024)

2.3.3. Evaluación de CEPs

La evaluación de las Características Emprendedoras Personales (CEPs) es un proceso crítico para identificar y fortalecer los atributos esenciales que potencian el desempeño emprendedor. Estas evaluaciones ayudan a medir competencias específicas como la autogestión, la creatividad, el liderazgo, la orientación a resultados, y otras habilidades fundamentales para la actividad empresarial. En México, el uso de herramientas psicométricas, cuestionarios estandarizados y entrevistas estructuradas se ha incrementado en instituciones educativas, centros de innovación y programas de desarrollo emprendedor (Rafael Eduardo Saavedra Leyva, 2022).

Métodos de Evaluación

a) Cuestionarios de Autoevaluación

Los cuestionarios de autoevaluación son herramientas diseñadas para que los individuos analicen sus propias fortalezas y debilidades en términos de habilidades emprendedoras. Estas evaluaciones, basadas comúnmente en la escala de Likert, solicitan a los participantes calificar su desempeño en áreas clave como:

- *Planificación estratégica:* Capacidad para diseñar metas y objetivos claros.
- *Toma de decisiones:* Evaluación de la habilidad para decidir bajo incertidumbre.
- *Motivación personal:* Nivel de compromiso hacia metas a largo plazo (Javier Suárez-Álvarez, 2016).

b) Pruebas Psicométricas

Las pruebas psicométricas buscan evaluar de manera objetiva rasgos como la resiliencia, la creatividad, la capacidad de asumir riesgos y el pensamiento crítico. Estas herramientas son

particularmente útiles para identificar áreas de mejora y potencial en aspirantes a programas de emprendimiento.

En México, universidades como el Tecnológico Nacional de México y el ITAM utilizan pruebas como el Big Five Personality Traits adaptadas a contextos locales para medir las CEPs. Estas pruebas también se aplican en concursos y programas de aceleración de empresas, como los promovidos por la Secretaría de Economía (coneval, s.f.) (Pública, 2015).

c) Entrevistas Estructuradas

Las entrevistas estructuradas permiten evaluar aspectos cualitativos de las CEPs, como la iniciativa, la comunicación efectiva y la resolución de problemas. En estas entrevistas, los evaluadores emplean preguntas diseñadas para explorar cómo los individuos enfrentan retos, gestionan recursos y lideran equipos. En México, este método se implementa en programas de formación emprendedora organizados por universidades y cámaras de comercio, donde los emprendedores presentan proyectos y reciben retroalimentación para mejorar sus capacidades (coneval, s.f.).

a) Importancia del Factor de Corrección en la Evaluación de CEPs

El factor de corrección es un elemento clave en la evaluación de las características emprendedoras personales (CEPs), ya que nos va a permitir ajustar y equilibrar los resultados obtenidos en los cuestionarios y las pruebas psicométricas. Su propósito principal es reducir los sesgos derivados de la autopercepción de los evaluados, así nos garantiza una medición mas precisa y confiable de sus habilidades y competencias emprendedoras. En esta evaluación, los participantes pueden sobreestimar o subestimar sus capacidades esto debido a los factores como la falta de experiencia, el exceso de confianza o la influencia del contexto en el que se encuentran. Para corregir estos sesgos, algunos instrumentos han incorporado factores de ajuste en criterios estadísticos o en comparaciones con grupos de referencia. Universidades y organismos de México han implementado estos ajustes en sus herramientas de evaluación, así asegurándose que los diagnósticos son más precisos y útiles para el desarrollo de los emprendedores. (Características emprendedoras personales (CEP), 2013)

Instrumentos Utilizados

En el país, se han desarrollado y adaptado instrumentos internacionales para evaluar las CEPs. Entre los más utilizados se encuentran:

- **Test de Actitud Emprendedora (TAE):** Instrumento diseñado para medir la orientación emprendedora y la disposición para asumir riesgos.
- **Cuestionarios de Competencias Emprendedoras:** Diseñados para evaluar habilidades específicas como la negociación y el pensamiento estratégico.
- **Mapeo de Habilidades Emprendedoras:** Técnica que combina autoevaluaciones y entrevistas para identificar fortalezas y áreas de desarrollo en emprendedores (Javier Suárez-Álvarez, 2016).

El emprendimiento en México

En México, la evaluación de CEPs ha tomado relevancia en instituciones académicas y programas gubernamentales. Ejemplos destacados incluyen:

- ***Tecnológico Nacional de México (TecNM):*** *Sus programas de incubación integran la evaluación de CEPs como un requisito para acceder al apoyo de mentores y recursos financieros (coneval, s.f.).*
- ***Instituto Nacional del Emprendedor (INADEM):*** *Promovió metodologías para medir y desarrollar competencias emprendedoras entre jóvenes y pequeños empresarios, especialmente en sectores rurales y comunidades indígenas (Pública, 2015).*

Además, en estados como Campeche, se han implementado programas regionales enfocados en medir las CEPs en sectores como el turismo y la agroindustria. Estas evaluaciones han permitido identificar perfiles emprendedores con alto potencial para generar impacto económico y social (Pública, 2015).

Beneficios de la Evaluación

La evaluación de las CEPs no solo permite a los individuos conocer su nivel de desarrollo en habilidades clave, sino también a las instituciones diseñar programas personalizados para fortalecer las capacidades detectadas. Esto contribuye directamente al éxito de proyectos emprendedores al proporcionar herramientas y estrategias adaptadas a las necesidades y fortalezas individuales.

2.3.4. Impacto de las CEPs en el Éxito Emprendedor

Las CEPs tienen un impacto directo en el éxito de las iniciativas emprendedoras, ya que influyen en la capacidad de los individuos para adaptarse a los desafíos del mercado, liderar equipos y generar valor. En México, estudios realizados en PyMEs destacan que los emprendedores con un perfil fuerte en CEPs tienen mayores tasas de supervivencia empresarial, especialmente en sectores como tecnología, comercio y servicios (Segura, 2019).

Factores Clave

- ***Adaptación al cambio:*** *Las CEPs permiten a los emprendedores ajustarse rápidamente a los cambios económicos y tecnológicos.*
- ***Resiliencia:*** *Los emprendedores con altas capacidades de resiliencia pueden superar fracasos y aprender de ellos para mejorar sus estrategias futuras.*
- ***Creación de redes:*** *Las CEPs fomentan habilidades sociales que facilitan la creación de alianzas estratégicas y el acceso a recursos.*

Evidencia en México

En regiones como Campeche, donde el emprendimiento se enfrenta a desafíos como el acceso limitado a financiamiento y mercados, las CEPs han demostrado ser un diferenciador importante. Los emprendedores que desarrollan estas características son capaces de innovar en sectores tradicionales y generar un impacto positivo en sus comunidades (Reporte sobre la Complejidad Económica del Estado de Campeche, 2017)

2.4. METODOLOGÍA SCRUM

La metodología Scrum es un marco de trabajo ágil que es utilizado principalmente en la gestión y desarrollo de proyectos. Surge como una respuesta a la necesidad de adaptarse rápidamente a cambios y entregar valor continuo en proyectos complejos. Su naturaleza iterativa e incremental ha permitido su implementación en diversas industrias, destacando en el desarrollo de software.

2.4.1. Introducción a Scrum

Scrum fue desarrollado por Ken Schwaber y Jeff Sutherland en la década de 1990 como parte del Manifiesto Ágil, buscando optimizar la colaboración entre equipos multidisciplinarios. En Scrum, el trabajo se organiza en ciclos cortos y repetitivos llamados sprints, lo que permite evaluar y ajustar resultados continuamente. Esta metodología se basa en tres pilares fundamentales:

- **Transparencia:** Todos los aspectos significativos del proceso deben ser visibles para los interesados.
- **Inspección:** Se realiza un monitoreo frecuente para identificar problemas y oportunidades de mejora.
- **Adaptación:** Permite ajustar estrategias en función de los hallazgos durante el desarrollo.

En México, su uso ha crecido significativamente en áreas como la tecnología y la educación, debido a su capacidad para fomentar la colaboración y adaptarse a contextos cambiantes (Schwaber, 2020) (1, 2021).

2.4.2. Componentes de Scrum

La metodología Scrum está estructurada en componentes clave que facilitan su implementación efectiva y aseguran la colaboración entre los involucrados. Estos elementos se dividen en roles, artefactos, y eventos, cada uno con funciones específicas para el éxito del proyecto (Schwaber, 2020).

1. Roles en Scrum

Los roles dentro de Scrum son esenciales para garantizar una clara distribución de responsabilidades y fomentar la sinergia entre los miembros del equipo. Cada rol tiene objetivos específicos que contribuyen al desarrollo continuo y alineado con las expectativas del cliente.

- **Product Owner:** Es la persona encargada de construir estrategias para aumentar el valor del producto. Este rol implica gestionar el Product Backlog, que contiene los requisitos y características del producto, priorizando los elementos según su valor para

el cliente o usuario final. Además, el Product Owner actúa como enlace directo entre el equipo de desarrollo y las partes interesadas, asegurándose de que los objetivos estén alineados con las necesidades del negocio.

- **Scrum Master:** Este rol es el facilitador del proceso Scrum. Su principal función es garantizar que el equipo entienda y siga los principios ágiles, ayudando a eliminar impedimentos que puedan surgir durante el desarrollo. El Scrum Master también fomenta una cultura de mejora continua, supervisando las dinámicas del equipo y asegurándose de que las reuniones y prácticas Scrum se lleven a cabo de manera efectiva.
- **Equipo de Desarrollo:** Se compone de profesionales multidisciplinarios encargados de convertir las ideas y requisitos en incrementos funcionales del producto. Este equipo es autónomo y colabora activamente para alcanzar los objetivos del sprint. Cada miembro es responsable de entregar resultados de alta calidad, trabajando en un entorno flexible y dinámico.

2. Artefactos de Scrum

- **Product Backlog:** Lista dinámica que contiene todas las funcionalidades, requisitos y mejoras necesarias para el producto. Está en constante actualización según las necesidades del cliente y del proyecto.
- **Sprint Backlog:** Subconjunto del Product Backlog que el equipo se compromete a desarrollar durante un sprint. Incluye tareas específicas y objetivos claros.
- **Incremento:** Representa el resultado funcional y utilizable al final de cada sprint, que debe cumplir con los criterios de aceptación definidos.

3. Eventos de Scrum

- **Sprint Planning:** Reunión inicial de cada sprint, donde el equipo define los objetivos y planifica las tareas a realizar.
- **Daily Scrum:** Reuniones diarias de 15 minutos para que el equipo comparta avances, obstáculos y planes para el día.
- **Sprint Review:** Al finalizar el sprint, el equipo presenta el trabajo completado a las partes interesadas para recibir retroalimentación.
- **Sprint Retrospective:** Reunión para reflexionar sobre el proceso y proponer mejoras en la dinámica del equipo.

2.4.3. Proceso de Implementación de Scrum

El proceso de implementación de Scrum sigue una estructura iterativa e incremental que permite a los equipos trabajar de manera organizada y eficiente. A continuación, se detallan los pasos clave:

1. Creación del Product Backlog

El proceso comienza con la recopilación de los requisitos del producto o proyecto. Estos requisitos, definidos como historias de usuario o tareas específicas, se priorizan en una lista dinámica conocida como el Product Backlog. Este documento es gestionado por el Product Owner, quien asegura que las necesidades del cliente estén alineadas con los objetivos del proyecto.

2. Planificación del Sprint

En esta etapa, el equipo selecciona del Product Backlog los elementos que se trabajarán durante el sprint, tomando en cuenta el tiempo y los recursos disponibles. Se define un Sprint Goal, que actúa como el objetivo principal del ciclo. Esta planificación asegura que el equipo esté enfocado en tareas específicas y alcanzables.

3. Ejecución del Sprint

Durante el sprint, el equipo de desarrollo trabaja en los elementos seleccionados para convertirlos en incrementos funcionales. Para mantener la coordinación, se llevan a cabo reuniones diarias de corta duración conocidas como Daily Scrums, donde los miembros comparten avances, identifican obstáculos y ajustan tareas según sea necesario.

4. Revisión del Sprint

Al finalizar el sprint, los resultados se presentan a los interesados en una reunión denominada Sprint Review. Aquí se evalúa el incremento entregado, se recibe retroalimentación y se discuten posibles ajustes o nuevas prioridades para el próximo ciclo.

5. Retrospectiva del Sprint

Finalmente, el equipo reflexiona sobre el proceso en la Sprint Retrospective. Este paso es crucial para identificar fortalezas, debilidades y oportunidades de mejora, asegurando que los aprendizajes obtenidos se implementen en futuros sprints.

En México, empresas líderes en tecnología como Softtek y Neoris han adoptado Scrum para desarrollar soluciones innovadoras, mejorando tiempos de entrega y satisfacción del cliente. Por otro lado, universidades como el Tecnológico Nacional de México han incorporado esta metodología en proyectos educativos y de investigación, fomentando en los estudiantes habilidades de colaboración, adaptabilidad y enfoque en resultados (Violeta Martínez Ramírez, 2021) (Ibero MTY, s.f.).

Scrum no solo optimiza los procesos de desarrollo, sino que también crea un ambiente de trabajo dinámico donde el equipo aprende y mejora continuamente, adaptándose a los cambios del entorno y asegurando la entrega de valor en cada iteración.

2.4.4. Ventajas y Desafíos de Scrum

Scrum ofrece un marco ágil que proporciona múltiples beneficios para la gestión de proyectos, pero también presenta retos que deben ser gestionados adecuadamente para garantizar su éxito (Schwaber, 2020).

Ventajas de Scrum

- **Flexibilidad:** Scrum permite adaptarse rápidamente a los cambios en los requisitos, gracias a su enfoque iterativo e incremental. Esto es especialmente útil en entornos dinámicos donde las necesidades pueden evolucionar constantemente.
- **Colaboración:** La metodología fomenta la comunicación continua entre el equipo de desarrollo, el Product Owner y otras partes interesadas. Las reuniones frecuentes, como el Daily Scrum, aseguran que todos estén alineados y trabajen hacia un objetivo común.
- **Entrega de Valor Constante:** Con la entrega de incrementos funcionales al final de cada sprint, los clientes reciben valor tangible de manera frecuente, lo que aumenta la satisfacción y facilita la retroalimentación inmediata.
- **Fomento de la Innovación:** Scrum otorga autonomía a los equipos para proponer y experimentar con soluciones creativas, promoviendo la innovación y un enfoque centrado en resultados.
- **Enfoque en la Mejora Continua:** A través de las retrospectivas al final de cada sprint, los equipos identifican áreas de mejora, optimizando tanto su rendimiento como la calidad de los entregables.

Desafíos de Scrum

- **Resistencia al Cambio:** Muchas organizaciones enfrentan barreras culturales al implementar Scrum, especialmente aquellas que están acostumbradas a modelos tradicionales de gestión. La resistencia puede manifestarse en la falta de compromiso o en la incomprensión de los roles ágiles.
- **Dependencia del Equipo:** Para tener éxito en la metodología Scrum depende en gran medida de la capacidad y compromiso del equipo. Equipos no entrenados o con baja cohesión pueden enfrentar dificultades para implementar el marco de manera efectiva.
- **Manejo del Alcance:** La flexibilidad de Scrum puede convertirse en un problema si no se gestionan adecuadamente las prioridades. La acumulación de tareas no esenciales puede desviar el enfoque de los objetivos principales del proyecto.
- **Requiere un Cambio de Mentalidad:** Scrum no solo implica nuevos procesos, sino también un cambio en la forma de pensar. Para muchas organizaciones, pasar de un modelo jerárquico a uno colaborativo puede ser un desafío significativo.

2.5. ESCALA LINKERT

La Escala Likert es una herramienta ampliamente utilizada en investigaciones sociales, educativas y psicológicas para poder medir actitudes, opiniones y percepciones. Fue desarrollada por el psicólogo estadounidense Rensis Likert en 1932, esta escala nos permite a los encuestados expresar nuestro grado de acuerdo o desacuerdo con una serie de afirmaciones, así facilitando así la cuantificación de las variables subjetivas. Ante de su desarrollo, las técnicas existentes eran mas complejas y menos accesibles para los investigadores. (Matas, s.f.)

2.5.1. Estructura y funcionamiento.

Una Escala Likert típica presenta una serie de ítems o afirmaciones relacionadas con el tema de estudio. Los encuestados indican su nivel de acuerdo o desacuerdo utilizando una escala ordinal, que comúnmente consta de cinco puntos:

- 1. Totalmente en desacuerdo*
- 2. En desacuerdo*
- 3. Neutral*
- 4. De acuerdo*
- 5. Totalmente de acuerdo*

Sin embargo, el número de puntos puede variar, siendo también comunes las escalas de siete o nueve puntos. Es importante que los ítems sean claros y específicos, y que la escala de respuesta sea equilibrada, así nos ofrece opciones tanto positivas como negativas. Además, que es recomendable incluir una opción neutral para poder capturar la indiferencia o la falta de opinión del encuestado. (Matas, s.f.)

2.5.2. Aplicaciones en investigación.

La Escala Likert es una herramienta esencial en diversas áreas de investigación debido a su versatilidad y su facilidad de uso. En el ámbito educativo, se emplea para evaluar percepciones de los estudiantes y docentes sobre los métodos de enseñanza. En la psicología, es útil para poder medir las actitudes y creencias sobre diversos temas. En el mundo empresarial, se utiliza para poder sondear la satisfacción del cliente. Un claro ejemplo en los estudios sobre las características emprendedoras personales (CEPs), la escala nos permite evaluar atributos como la autoconfianza y la capacidad de asumir los riesgos. (Matas, s.f.)

2.5.3. Ventajas y desventajas.

Gracias a los distintos grados de respuesta de la escala Likert nos permite conocer la postura de las personas con respecto a algo o alguien. Estas son algunas de sus ventajas y desventajas (Muguira, 2024):

Ventajas:

- **Simplicidad:** *Fácil de diseñar, administrar y analizar.*
- **Flexibilidad:** *Aplicable en una amplia gama de contextos y temas.*
- **Sensibilidad:** *Capta matices en las opiniones de los encuestados, proporcionando datos más detallados.*

Desventajas:

- **Sesgo de aquiescencia:** *Existen casos en donde los encuestados tienden a estar de acuerdo con las afirmaciones independientemente de su contenido.*

- **Efecto de deseabilidad social:** Los participantes pueden responder de manera que creen que es socialmente aceptable en lugar de ser sinceros.
- **Limitación en la profundidad:** Aunque proporciona una visión general de las actitudes, no profundiza en las razones subyacentes de dichas actitudes.

2.5.4. Métodos de análisis de datos.

Los datos recopilados mediante las Escalas Likert se analizan generalmente utilizando técnicas estadísticas descriptivas e inferenciales. Es común calcular medias y desviaciones estándar para cada uno de los ítems, también se pueden realizar análisis de confiabilidad, como el coeficiente alfa de Cronbach, para poder evaluar la consistencia interna de la escala. Además, se puede emplear análisis factoriales para poder identificar dimensiones subyacentes en los datos. Sin embargo, es importante tener en cuenta que, aunque las respuestas en una Escala Likert son ordinales, muchos de los investigadores tratan estos datos como intervalos para facilitar el análisis estadístico. (Matas, s.f.)

2.6. PERFIL DE EGRESO

El Perfil de Egreso de un alumno es una descripción detallada de las competencias, conocimientos, habilidades y actitudes que debe haber adquirido al finalizar su programa educativo. Este perfil es fundamental en la educación superior, ya que establece los lineamientos que nos garantizan la formación integral de los egresados y su adecuada entrada en el ámbito profesional y social. En México, las universidades y los organismos educativos definen estos perfiles con base en los requerimientos del mercado laboral, las necesidades del sector productivo y las políticas nacionales de educación superior. (Claudia García Ancira, 2020)

2.6.1. Importancia del Perfil de Egreso.

El perfil de Egreso no solo define las competencias técnicas de un profesional, sino también sus capacidades transversales, como el pensamiento crítico, la resolución de problemas y la comunicación efectiva. Este perfil es un referente clave para la acreditación de programas académicos y para la certificación de competencias. Además, es un elemento importante en el diseño curricular, ya que nos permite evaluar si los contenidos y las estrategias de enseñanza cumplen con los objetivos educativos planteados (Ramos-Peña, 2018).

2.6.2. Componentes del Perfil de Egreso.

El perfil se estructura en diversas dimensiones que garantizan la formación integral de los estudiantes (URUGUAY, 2021):

a) Competencias Disciplinarias:

- Dominio de los conocimientos teóricos y prácticos específicos del área de estudio.
- Capacidad para aplicar metodologías y herramientas propias de la disciplina.
- Habilidad para desarrollar proyectos innovadores y contribuir al avance del conocimiento.

b) Competencias Genéricas:

- *Pensamiento crítico y reflexivo para la toma de decisiones fundamentadas.*
- *Adquirir la habilidad para el trabajo en equipo y la colaboración interdisciplinaria.*
- *Desarrollo de habilidades de comunicación oral y escrita.*
- *Adaptabilidad al cambio y aprendizaje continuo.*

c) Valores y Actitudes:

- *Responsabilidad ética y compromiso social.*
- *Actitud emprendedora e iniciativa para la innovación.*
- *Conciencia ambiental y desarrollo sostenible.*

2.6.3. Diseño del Perfil de Egreso en México.

*En México, el perfil de egreso se establece en base a las normativas y modelos educativos promovidos por la **secretaría de Educación Pública (SEP)** y organismos como la **Asociación Nacional de Universidades e Instituciones de Educación Superior (ANUIES)**. Además, se considera el **Marco Mexicano de Cualificaciones** para asegurar la alineación con los estándares internacionales. Las universidades suelen definir los perfiles de egreso mediante consultas a expertos, análisis de tendencias laborales y las evaluaciones de necesidades del sector productivo. Un claro ejemplo de esto es el **Modelo Educativo 2030**, que busca formar egresados con enfoque humanista, capacidades digitales y responsabilidad social ((SEP), 2022).*

2.6.4. Relación con el Mercado Laboral.

El perfil de egreso de cada programa educativo debe de responder a las demandas del mercado laboral, garantizando que los profesionales cuenten con las competencias necesarias para desempeñarse exitosamente. En este sentido, las universidades han forzado la enseñanza de habilidades como (Omar Alejandro Pérez Cruz, 2020):

- *Dominio de tecnologías digitales.*
- *Gestión del conocimiento y resolución de problemas complejos.*
- *Emprendimiento e innovación.*
- *Competencias interculturales y dominio de idiomas.*

2.6.5. Evaluación y Seguimiento del Perfil de Egreso.

Las instituciones de educación superior han implementado mecanismos de evaluación para verificar que el perfil de egreso se cumpla en la práctica. Algunas de las estrategias implementadas incluyen ((COPAES), 2019):

- *Encuestas a egresados y empleadores.*
- *Evaluaciones de desempeño profesional.*
- *Análisis de inserción laboral y satisfacción de egresados.*

2.7. ESTUDIO LONGITUDINAL

Este método de investigación que nos permite analizar cambios y las tendencias a lo largo del tiempo en una población específica. Está basado en la recopilación de datos en diferentes momentos, lo que nos facilita a la identificación de patrones de evolución en variables como el comportamiento, el rendimiento académico, la salud o el desarrollo profesional. En México, este tipo de estudios han sido ampliamente utilizados en áreas como la educación, la salud pública y el análisis del emprendimiento, permitiéndonos evaluar políticas y programas con base en la evolución de los individuos o grupos estudiados (Team, 2023).

2.7.1. Características del Estudio longitudinal.

Los estudios longitudinales presentan las siguientes características (¿Quees?, 2023):

- **Observación a lo largo del tiempo:** Se examina a los mismos sujetos en diferentes periodos.
- **Análisis de cambios y tendencias:** Permiten identificar la evolución de un fenómeno en función del tiempo.
- **Mayor precisión en la causalidad:** A diferencia de estudios transversales, los longitudinales ayudan a establecer relaciones causa-efecto.
- **Uso de mediciones repetidas:** Se aplican instrumentos de evaluación en distintas etapas para comparar resultados.

2.7.2. Tipos de estudios longitudinales.

Existen diversas clasificaciones de los estudios longitudinales según su enfoque y aplicación (Team, 2023):

a) Estudios de cohorte

Este estudio se realiza a un grupo de individuos que comparten una característica en común. (por ejemplo, haber egresado de una universidad en el mismo año) y analiza su desarrollo a lo largo del tiempo. En México, se han realizado estudios de cohorte para evaluar la empleabilidad y desarrollo profesional de egresados universitarios.

b) Estudios de panel

Los estudios de panel analizan a los mismos sujetos en diferentes puntos del tiempo generalmente con encuestas o entrevistas periódicas. Un ejemplo en México es la **Encuesta Nacional de Ocupación y Empleo (ENOE)**, que sigue a hogares y trabajadores en distintos momentos para evaluar su inserción en el mercado laboral.

c) Estudios retrospectivos

Este enfoque se basa en la recopilación de información sobre experiencias pasadas de los sujetos. Aunque no es completamente longitudinal, nos permite identificar tendencias y trayectorias a partir de datos históricos.

2.7.3. Aplicaciones del Estudio Longitudinal.

En México, los estudios longitudinales han sido clave en diversas áreas (Sybing, 2025):

1. Educación y Trayectoria Laboral

- *Evaluación del impacto de programas educativos en el desempeño profesional.*
- *Seguimiento de egresados para conocer su inserción en el mercado laboral y su nivel de ingresos.*
- *Análisis de la evolución de competencias emprendedoras en estudiantes universitarios.*

2. Salud Pública

- *Estudios sobre enfermedades crónicas y sus factores de riesgo a lo largo del tiempo.*
- *Evaluación del impacto de políticas de salud en poblaciones vulnerables.*

3. Emprendimiento y Economía

- *Seguimiento de empresas emergentes para analizar su crecimiento y sostenibilidad.*
- *Evaluación de políticas de financiamiento y su impacto en el desarrollo de negocios.*

2.7.4. Ventajas y Desafíos del Estudio Longitudinal.

Las ventajas y desventajas del estudio longitudinal sobre los métodos de investigación incluyen los siguientes (Castillo, 2022):

Ventajas:

- *Permite observar cambios en individuos o grupos a lo largo del tiempo.*
- *Facilita el análisis de relaciones causa-efecto con mayor precisión.*
- *Proporciona datos valiosos para la toma de decisiones en políticas públicas y estrategias empresariales.*

Desventajas.

- *Alto costo y requerimiento de tiempo prolongado para su ejecución.*
- *Posible pérdida de participantes a lo largo del estudio (deserción de muestra).*
- *Complejidad en el análisis y gestión de grandes volúmenes de datos.*

2.7.5. Estudios Longitudinales en México.

En México, diversas instituciones han implementado estudios longitudinales para evaluar impactos en educación, empleo y salud. Algunos de los ejemplos son:

- **Estudio de Trayectoria de Egresados Universitarios** (ANUIES, 2021): *Analiza la empleabilidad y desempeño de egresados en distintas generaciones.*
- **Encuesta Nacional de Salud y Nutrición** (INEGI, 2022): *Evalúa la evolución de la salud en la población mexicana a lo largo del tiempo.*
- **Proyecto GEM México** ((GEM), 2021): *Realiza un seguimiento de las tendencias del emprendimiento y los factores que influyen en el éxito empresarial.*

Capítulo 3

3.1 MATERIALES Y MÉTODOS.

En este capítulo se describen los materiales y métodos utilizados para el desarrollo del proyecto, detallando las herramientas tecnológicas empleadas y las técnicas aplicadas durante las fases de implementación. El enfoque principal fue garantizar un flujo de trabajo ágil y eficiente, permitiendo que cada una de las etapas del proyecto pueda replicarse en otros contextos con las mismas condiciones.

Durante el desarrollo del sistema de inteligencia de negocios, se emplearon diversas herramientas de software y lenguajes de programación que permitieron estructurar y consolidar el proyecto. A continuación, se describen las principales tecnologías y sus funcionalidades en el contexto del proyecto:

3.1.1 RECURSOS SOFTWARE

3.1.2.1 *Angular*

Angular es un framework de desarrollo de aplicaciones web de código abierto mantenido por Google, que permite crear aplicaciones dinámicas de una sola página (SPA, por sus siglas en inglés). En el proyecto, Angular fue utilizado para desarrollar el frontend del sistema, lo que permitió la creación de interfaces interactivas y responsivas.

Entre las ventajas de Angular destacan:

- **Desarrollo estructurado:** Al usar TypeScript y seguir patrones de desarrollo modular, Angular facilita el mantenimiento del código.
- **Data binding:** Se implementó el enlazado de datos bidireccional para mantener la sincronización entre la vista y el modelo de datos.
- **Componentes reutilizables:** El uso de componentes permitió estructurar la aplicación en pequeñas piezas funcionales, mejorando la reutilización de código.

3.1.2.2 *Laravel*

Laravel es un framework de PHP que facilita el desarrollo de aplicaciones web siguiendo el patrón MVC (Modelo-Vista-Controlador). Laravel fue utilizado para gestionar el backend del sistema, proporcionando una plataforma sólida para la gestión de usuarios, grupos y cuestionarios.

Las características más importantes de Laravel en este proyecto incluyen:

- **Rutas y controladores:** Se configuraron rutas que permiten la interacción entre el frontend y las APIs del sistema.
- **Migrations y Eloquent ORM:** La estructura de la base de datos fue gestionada mediante migraciones, y las relaciones entre las tablas fueron manejadas a través de Eloquent ORM.
- **Middleware de autenticación:** Laravel ofreció soporte para la implementación de la autenticación de usuarios mediante middleware.

3.1.2.3 MySQL

MySQL fue el sistema de gestión de bases de datos utilizado para almacenar la información del sistema. Se seleccionó MySQL por su capacidad para manejar grandes volúmenes de datos de manera eficiente y su fácil integración con Laravel.

Entre los aspectos más relevantes de MySQL en el proyecto se encuentran:

- **Gestión de tablas:** Se crearon tablas para almacenar la información de usuarios, cuestionarios, grupos, entre otros.
- **Relaciones entre tablas:** La base de datos fue diseñada con relaciones clave foránea para asegurar la integridad referencial entre los distintos módulos.

3.1.2.4 Git Hub

GitHub fue utilizado como plataforma de control de versiones, permitiendo la colaboración y el manejo del código fuente de manera distribuida. El uso de GitHub en el proyecto permitió:

- **Control de versiones:** Se mantuvo un historial claro de los cambios realizados en el código a lo largo del tiempo.
- **Colaboración:** La gestión de ramas facilitó el trabajo colaborativo y la integración continua entre el frontend y el backend.

3.1.2.5 JSON Web Tokens

JWT fue utilizado para la autenticación de los usuarios en el sistema. Este estándar asegura la transferencia segura de información entre el cliente y el servidor mediante la creación de tokens firmados digitalmente.

Entre las ventajas del uso de JWT destacan:

- **Autenticación basada en tokens:** Los tokens generados se utilizan para identificar al usuario en cada solicitud al servidor, permitiendo la autenticación sin necesidad de gestionar sesiones.
- **Seguridad:** El uso de firmas en los tokens asegura la integridad y autenticidad de la información transferida.

3.1.2.6 Lenguaje de programación

JavaScript

JavaScript es un lenguaje de programación interpretado y orientado a objetos, ampliamente utilizado para el desarrollo de aplicaciones web. En este proyecto, JavaScript fue utilizado en el frontend, principalmente en el desarrollo de las interacciones y la lógica de los componentes de Angular.

Las principales características de JavaScript que se aprovecharon incluyen:

- **Manipulación del DOM:** Se utilizó para actualizar la interfaz de usuario de manera dinámica sin necesidad de recargar la página.

- **Funciones asíncronas:** El manejo de promesas y `async/await` permitió gestionar peticiones a la API de manera eficiente.

TypeScript

TypeScript es un lenguaje de programación desarrollado por Microsoft que extiende la funcionalidad de JavaScript, añadiendo tipado estático opcional. En este proyecto, TypeScript fue utilizado en el desarrollo del frontend mediante Angular, facilitando la creación de componentes robustos y escalables.

Entre las características más destacadas de TypeScript en este proyecto se encuentran:

- **Tipado estático:** Permite definir tipos para variables, funciones y objetos, lo que ayuda a evitar errores durante el tiempo de desarrollo y hace que el código sea más predecible y fácil de mantener.
- **Compatibilidad con JavaScript:** Al compilarse a JavaScript puro, TypeScript es completamente compatible con cualquier entorno que soporte JavaScript.
- **Sophorte para herramientas modernas:** TypeScript cuenta con herramientas avanzadas de depuración y autocompletado, lo que agiliza el desarrollo en Angular.

PHP

PHP es el lenguaje de programación utilizado en el backend, específicamente con el framework Laravel. Su capacidad para gestionar la lógica del servidor y su integración con bases de datos lo convierte en una excelente opción para el desarrollo de aplicaciones web robustas.

Las principales características de PHP que se aprovecharon en el proyecto incluyen:

- **Desarrollo basado en servidores:** PHP procesa las solicitudes del frontend, interactúa con la base de datos MySQL y devuelve los datos necesarios para las operaciones del sistema.
- **Integración con Laravel:** Laravel, al estar basado en PHP, permite una estructuración clara de la aplicación mediante el uso del patrón MVC (Modelo-Vista-Controlador).
- **Facilidad de uso y flexibilidad:** PHP ofrece una curva de aprendizaje accesible, facilitando la implementación rápida de soluciones en el backend.

3.2 DIAGRAMA DE BLOQUES DE LAS ETAPAS DEL PROYECTO

En este capítulo se presenta el diagrama de bloques que describe las principales etapas del desarrollo del sistema de inteligencia de negocios para la identificación y mejora de perfiles emprendedores. Este diagrama permite visualizar de manera clara y ordenada el flujo de trabajo desde el inicio del proyecto hasta su conclusión, y muestra cómo se interrelacionan las diferentes fases de este.

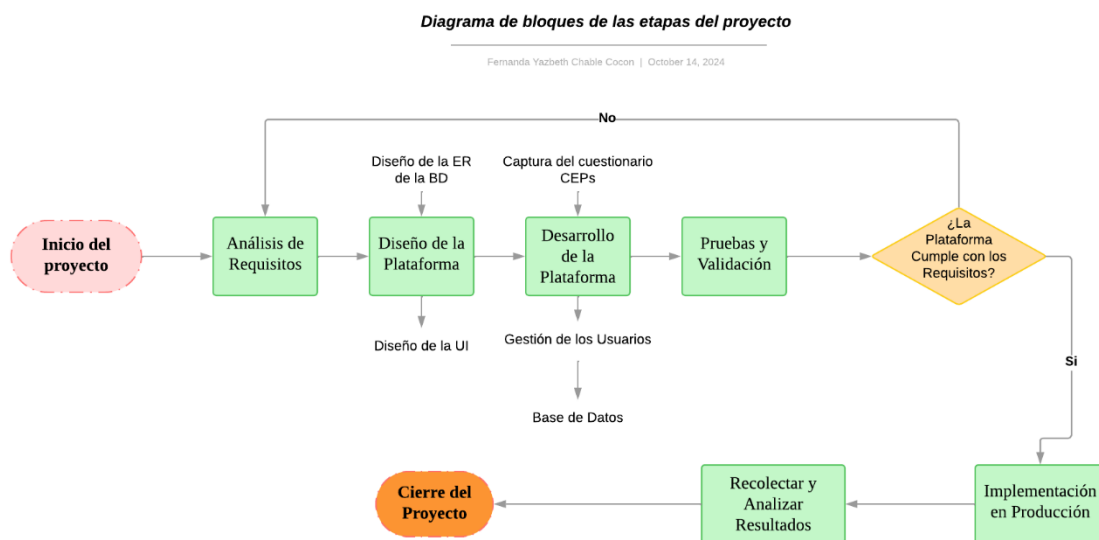


Ilustración 3. 1 Diagrama de etapas del proyecto

3.3 IMPLEMENTACIÓN DE LA METODOLOGÍA DEL PROYECTO

La metodología elegida para el desarrollo de este proyecto es Scrum, que nos permite gestionar el proceso de desarrollo de manera ágil y flexible, asegurando la entrega continua de valor en cada iteración. Esta metodología favorece la colaboración y la comunicación constante entre los miembros del equipo, lo que resulta fundamental para el éxito del proyecto.

A lo largo del desarrollo, se realizaron reuniones diarias de seguimiento, conocidas como "scrum diario", donde cada miembro del equipo compartía el progreso realizado, los obstáculos encontrados y las tareas a realizar. Estas reuniones permitieron una rápida identificación y resolución de problemas, además de mantener a todos alineados con los objetivos del proyecto.

Antes de iniciar con el levantamiento de los requerimientos, fue necesario definir claramente los roles del equipo Scrum. En la tabla 1 se detallan los miembros del equipo y sus responsabilidades.

Roles	Descripción	Persona
Product Owner	Coordinador del proyecto	Dra. Irene Alvarado Van der Laat
Scrum Master	Responsable del proyecto	Dra. Marlene Mendez Moreno
Development Team	Desarrollador del proyecto	Br. Fernanda Yazbeth Chable Cocon Br. Fernando Yahir Puc Euan

Tabla 3. 1 Equipo Scrum

Para el levantamiento de los requerimientos, se llevaron a cabo varias reuniones con los interesados del proyecto. Estas reuniones fueron esenciales para comprender a fondo las necesidades y expectativas de los usuarios finales, lo que nos permitió establecer los requerimientos específicos del sistema.

Los procesos que se contemplan en la aplicación incluyen:

Gestión de Usuarios:

- Registro de usuarios (estudiantes, maestros y administrativos) con información básica.
- Autenticación y recuperación de contraseña.

Gestión de Cuestionarios:

- Asignación y configuración de cuestionarios por parte de los maestros.
- Respuestas y retroalimentación inmediata para los estudiantes.

Gestión de Grupos:

- Creación y gestión de grupos de estudiantes por parte de los maestros.
- Asignación de estudiantes a grupos específicos.
- Creación del código de acceso a los grupos por parte de los maestros.

Análisis y Reportes:

- Generación de informes detallados sobre los resultados de los cuestionarios.
- Visualización de datos a través de gráficos y estadísticas.

TIPO DE USUARIOS	DESCRIPCIÓN
Estudiantes	Usuario que responde a los cuestionarios y visualiza su progreso académico.
Maestros	Encargado de asignar cuestionarios, visualizar los resultados de sus alumnos, crear grupos, administrar a los alumnos y generar informes.
Administrativos	Responsable de gestionar usuarios, categorías de preguntas, análisis de datos, editar y publicar los cuestionarios, agregar instituciones, editar la página principal de la plataforma, generar informes en base a los resultados de los alumnos.

Tabla 3. 2 Tipos de usuarios del sistema

3.4 REQUERIMIENTOS

En esta sección se detallaron los requerimientos del sistema propuesto, organizados en dos categorías principales: requerimientos funcionales y requerimientos no funcionales. Los requerimientos funcionales describen las características y funciones específicas que el sistema debe ofrecer a sus usuarios, mientras que los requerimientos no funcionales se refieren a las propiedades y restricciones que debe cumplir el sistema en términos de rendimiento, usabilidad, seguridad y otras cualidades.

3.4.1 REQUERIMIENTOS FUNCIONALES

A continuación, se presentan los requerimientos funcionales organizados por módulos y submódulos:

ID	REQUERIMIENTOS FUNCIONALES
Módulo de Bienvenida	
R01	La página principal debe mostrar un menú con seis botones, que incluyen las siguientes opciones: Home, Quiénes somos, Contacto, Cuestionario, Registro e Inicio de sesión. Cada botón debe llevar al usuario a una ventana o sección correspondiente dentro del módulo de bienvenida.
R02	El botón "Home" debe mostrar información relacionada con el cuestionario que se va a realizar, explicando los detalles del proceso.
R03	El botón "Quiénes somos" debe mostrar la misión, visión y una breve historia del proyecto.
R04	El botón "Contacto" debe mostrar los siguientes detalles: nombre de la institución, dirección, número de teléfono, sitio web, y un mapa con las universidades que participan en el proyecto.
R05	El botón "Cuestionario" debe mostrar un mensaje de advertencia, indicando que el cuestionario solo se puede acceder después de registrarse e iniciar sesión en la plataforma.
R06	El botón "Registro" debe permitir que el usuario ingrese sus datos (nombre, apellidos, correo, etc.) para crear una cuenta.
R07	El botón "Inicio de sesión" debe permitir a los usuarios registrados ingresar usando su correo y contraseña.
Módulo de Inicio de Sesión	
R08	Poder acceder con correo electrónico y contraseña.
R09	Poder registrar nuevo usuario proporcionando información básica (Nombre, Apellidos, Institución, Matrícula, Ocupación, País, correo electrónico, contraseña).
R10	Permite recuperar contraseña a través de un enlace enviado por correo electrónico.
Módulo de Perfil de Usuario	
Submódulo Perfil de Estudiante:	
R11	Los estudiantes podrán registrarse y acceder a la plataforma utilizando un nombre de usuario y contraseña.
R12	Ver los cuestionarios asignados por sus profesores.
R13	Responder a los cuestionarios desde cualquier dispositivo con conexión a internet.
R14	Recibir retroalimentación personalizada sobre sus características empresariales.
R15	Ver su progreso individual a lo largo de la carrera en base a los resultados de los cuestionarios.
R16	Consultar sus datos de perfil.
Submódulo Perfil de Maestro	
R17	Los maestros podrán asignar cuestionarios a grupos específicos de alumnos, filtrando por carrera o semestre.
R18	Establecer fechas de inicio y fin para los cuestionarios.

R19	Ver las respuestas de los alumnos a los cuestionarios, tanto individualmente como en conjunto.
R20	Generar informes detallados de los resultados de los cuestionarios, incluyendo cualidades positivas y negativas del alumno, así como su retroalimentación por sección.
R21	Exportar los informes en formato PDF.
R22	Generar códigos de acceso únicos para cada grupo para controlar el acceso a los cuestionarios.
R23	Consultar sus datos de perfil.
Submódulo Perfil Administrativo	
R24	Los administrativos podrán crear, editar y eliminar usuarios de la plataforma (alumnos, profesores y administradores).
R25	Gestionar las categorías y variables de las preguntas.
R26	Visualizar los resultados de todas las encuestas realizadas en la plataforma.
R27	Generar informes detallados de los resultados, incluyendo gráficos y tablas que muestran las cualidades positivas y negativas individuales y grupales por carrera, grado y sección.
R28	Exportar los informes en formato PDF.
R29	Consultar los datos de perfil de todos los usuarios.
R30	Realizar un seguimiento del uso de la plataforma.
R31	Proporcionar retroalimentación personalizada a los alumnos en base a sus respuestas.
R32	Guardar todas las respuestas y la retroalimentación de los grupos para realizar un seguimiento del progreso individual y grupal a lo largo de la carrera.
Módulo de Gestión de Cuestionarios	
Submódulo Edición de Cuestionarios	
R33	Los administradores podrán editar preguntas y variables dentro de los cuestionarios, declarando si una variable influye positiva o negativamente en el resultado.
Submódulo Configuración del Cuestionario	
R34	Posibilidad de configurar el cuestionario para un grupo específico y establecer una fecha límite para completarlo.
Submódulo Asignación de Cuestionarios:	
R35	Los profesores podrán asignar cuestionarios a grupos completos.
Submódulo Código de Acceso	
R36	Los estudiantes recibirán un código de acceso para completar el cuestionario, garantizando que solo los estudiantes asignados puedan acceder.
Submódulo Respuestas de los Estudiantes	
R37	Los estudiantes deberán poder responder a los cuestionarios en línea.
Submódulo Retroalimentación Inmediata	
R38	Una vez enviado el cuestionario, los estudiantes deben recibir retroalimentación inmediata sobre sus respuestas, incluyendo cualidades o características positivas y negativas, así como recomendaciones para mejorar.
Submódulo Eliminación de Cuestionarios Completados	

R39	Los cuestionarios completados deben desaparecer de la vista del estudiante para evitar consultas posteriores.
Submódulo Cálculo de Resultados	
R40	Las preguntas tendrán una fórmula que se calcula a partir de la escala proporcionada por el administrador.
Submódulo Preguntas de Corrección	
R41	Las preguntas de corrección deben tener una fórmula independiente para aumentar la precisión del cuestionario.
Submódulo Aleatorización de Preguntas	
R42	Las preguntas deben poder cambiar de sentido (positivo y negativo) y ser aleatorizadas en cada cuestionario.
Módulo de Gestión de Grupos	
Submódulo Creación de Grupos	
R43	Los maestros deberán poder crear y gestionar grupos de estudiantes según su carrera y grado.
Submódulo Asignación a Grupos	
R44	Los estudiantes deberán poder ser asignados a grupos específicos mediante un código de acceso proporcionado por el maestro.
Módulo de Dashboard	
Submódulo Dashboard para Estudiantes	
R45	Visualización de todos los cuestionarios completados por el estudiante y su progreso académico.
Submódulo Dashboard para Maestros	
R46	Listado de cuestionarios y grupos gestionados por el maestro, y visualización de resultados individuales y grupales.
Submódulo Dashboard para Administrativos	
R47	Dashboard para Administrativos: Acceso a resultados y estadísticas de todos los cuestionarios aplicados, así como detalles de usuarios y administración.
Módulo de Análisis y Reportes	
Submódulo Gráficas de Resultados	
R48	Visualización gráfica de los resultados obtenidos en los cuestionarios.
Submódulo Análisis de Datos	
R49	El administrador podrá analizar los datos obtenidos de los alumnos en base a sus características empresariales y capacidades.
Submódulo Generación de Informes	
R50	La plataforma generará automáticamente informes detallados de los resultados de los cuestionarios.
Submódulo Exportación a PDF	
R51	Posibilidad de exportar la lista de resultados y alumnos en formato PDF.
Submódulo Exportación a PDF	
R52	Posibilidad de exportar las gráficas de resultados por alumno o de forma grupal en formato PDF.

Tabla 3. 3 Requerimientos funcionales

3.4.2 REQUERIMIENTOS NO FUNCIONALES

A continuación, se presentan los requerimientos no funcionales que el sistema debe cumplir:

ID	Requerimientos No Funcionales
RN01	La aplicación debe ser compatible con navegadores modernos y versiones actualizadas de Angular y Laravel.
RN02	La aplicación debe ser responsiva y funcionar correctamente en dispositivos móviles y de escritorio.
RN03	Los mensajes de error deben ser claros y comprensibles para los usuarios.
RN04	Se debe incluir un diseño de interfaz amigable y accesible para todos los usuarios.
RN05	El rendimiento de la aplicación debe ser óptimo, garantizando tiempos de carga rápidos y una experiencia fluida.
RN06	La seguridad de los datos debe ser prioritaria, incluyendo cifrado de contraseñas y medidas contra ataques comunes.
RN07	Se debe utilizar MySQL como sistema de gestión de bases de datos para el almacenamiento de datos.
RN08	La aplicación debe contar con una infraestructura de servidor adecuada para garantizar la disponibilidad y escalabilidad.
RN09	La metodología de desarrollo de software será Scrum, promoviendo la adaptabilidad y mejora continua.
RN10	El lenguaje de programación utilizado será PHP (para Laravel) y TypeScript (para Angular).
RN11	Se debe garantizar que la aplicación sea fácil de mantener y actualizar en el futuro.
RN12	Se debe contar con una conexión a Internet para acceder a la plataforma y realizar las funciones necesarias.

Tabla 3. 4 Requerimientos no funcionales

3.5. PROPUESTA PRELIMINAR DE DISEÑO Y FUNCIONALIDADES MODULO DE BIENVENIDA

Al acceder al módulo de bienvenida, el usuario verá una pantalla inicial con el logo de la aplicación, colores distintivos y el nombre de la plataforma en tipografía clara, reflejando la identidad visual del sistema. Desde esta pantalla se accede al menú principal, que contiene las siguientes opciones:

- **Home:** Explica brevemente el propósito del cuestionario y su importancia para los usuarios.
- **Quiénes somos:** Muestra información sobre la misión, visión e historia del proyecto, resaltando su vínculo con las instituciones participantes.
- **Contacto:** Incluye datos de contacto como dirección, teléfono y sitio web, además de un mapa con las universidades participantes.

- **Cuestionario:** Muestra un mensaje de advertencia indicando que el usuario debe registrarse e iniciar sesión para acceder al cuestionario.
- **Registro:** Permite a los nuevos usuarios crear una cuenta en la plataforma.
- **Inicio de sesión:** Facilita el acceso a los usuarios registrados mediante correo electrónico o matrícula y contraseña.

Al ingresar a la plataforma, el usuario será redirigido a la pantalla de inicio, ya sea que tenga o no sesión iniciada, lo que permite una navegación fluida y fácil acceso a las opciones del menú.



Ilustración 3. 2 Diseño del menú principal de la plataforma

Submódulo Home:

El submódulo Home en el Módulo de Bienvenida contiene los siguientes elementos visuales y funcionales para guiar al usuario en la plataforma:

- **Encabezado:** Muestra el nombre de la plataforma en tipografía clara y legible, acompañado del logo, resaltando la identidad visual del sistema. Incluye un botón que permite acceder al cuestionario directamente si el usuario está logueado; de lo contrario, se muestra un mensaje de restricción indicando que el usuario debe iniciar sesión para continuar.
- **Carrusel de imágenes:** Debajo del encabezado, se presenta un carrusel de imágenes relacionadas con el tema del proyecto, brindando una introducción visual al propósito de la plataforma.
- **Objetivo del cuestionario:** Una breve sección que explica el propósito del cuestionario y su relevancia para el usuario.
- **Mensaje de advertencia:** Informa al usuario que el cuestionario es solo una prueba y no posee valor absoluto, aclarando que se usa únicamente con fines orientativos.
- **Pie de página:** En la parte inferior, el pie de página muestra información sobre el proyecto, instituciones participantes, créditos, teléfonos y enlaces a las redes sociales del proyecto.



Ilustración 3. 3 Diseño del encabezado de la página principal



Ilustración 3. 4 Diseño del carrusel de fotos y objetivos



Ilustración 3. 5 Diseño de advertencia y pie de pagina

Submódulo Quienes somos:

El submódulo Quiénes Somos proporciona al usuario información detallada sobre la visión y misión del proyecto, así como su historia y relación con las instituciones participantes. Está organizado en las siguientes secciones:

- **Misión:** Describe el propósito fundamental del proyecto, enfatizando su contribución al desarrollo académico y profesional de los usuarios.
- **Visión:** Expone las metas a largo plazo del proyecto, destacando el impacto esperado en la comunidad educativa y en el perfil emprendedor de los estudiantes.
- **Historia del proyecto:** Breve narrativa que explica el origen, la motivación y el proceso de creación del proyecto, así como la colaboración con instituciones académicas.
- **Equipo:** Presenta a los miembros del equipo de desarrollo, incluyendo sus nombres, roles en el proyecto y, opcionalmente, una breve descripción de sus áreas de especialización o contribuciones clave.



Ilustración 3. 6 Diseño de la ventana Quienes somos



Ilustración 3. 7 Diseño de la ventaja misión y visión del proyecto



Ilustración 3. 8 Diseño de la ventana Equipo

Submódulo de Contacto:

El submódulo Contacto proporciona los datos de contacto y ubicación de las universidades que participan en el proyecto. Está organizado en las siguientes secciones:

- **Datos de contacto:** Incluye la dirección, número de teléfono y sitio web de cada universidad participante en el proyecto.
- **Mapa de ubicaciones:** Muestra un mapa interactivo que indica la ubicación de las universidades involucradas, facilitando una referencia visual para los usuarios interesados.

Este submódulo permite a los usuarios obtener información de contacto y localizar fácilmente las universidades asociadas al proyecto.



Ilustración 3. 9 Diseño de la ventana para contactos

Submódulo de Cuestionario CEPs:

El submódulo Cuestionario informa al usuario sobre el acceso restringido al cuestionario. Contiene los siguientes elementos:

- **Mensaje de advertencia:** Se muestra un mensaje claro que indica que solo los usuarios registrados y logueados pueden acceder al cuestionario.
- **Botón de acceso:** Un botón que, al hacer clic, redirige a los usuarios no autenticados a la página de registro o inicio de sesión, permitiendo un acceso rápido si cumplen con el requisito de autenticación.

Este submódulo asegura que solo los usuarios registrados puedan realizar el cuestionario, orientando a los demás a registrarse o iniciar sesión para acceder.



Ilustración 3. 10 Diseño para la ventana de bloqueo

MÓDULO DE REGISTRO

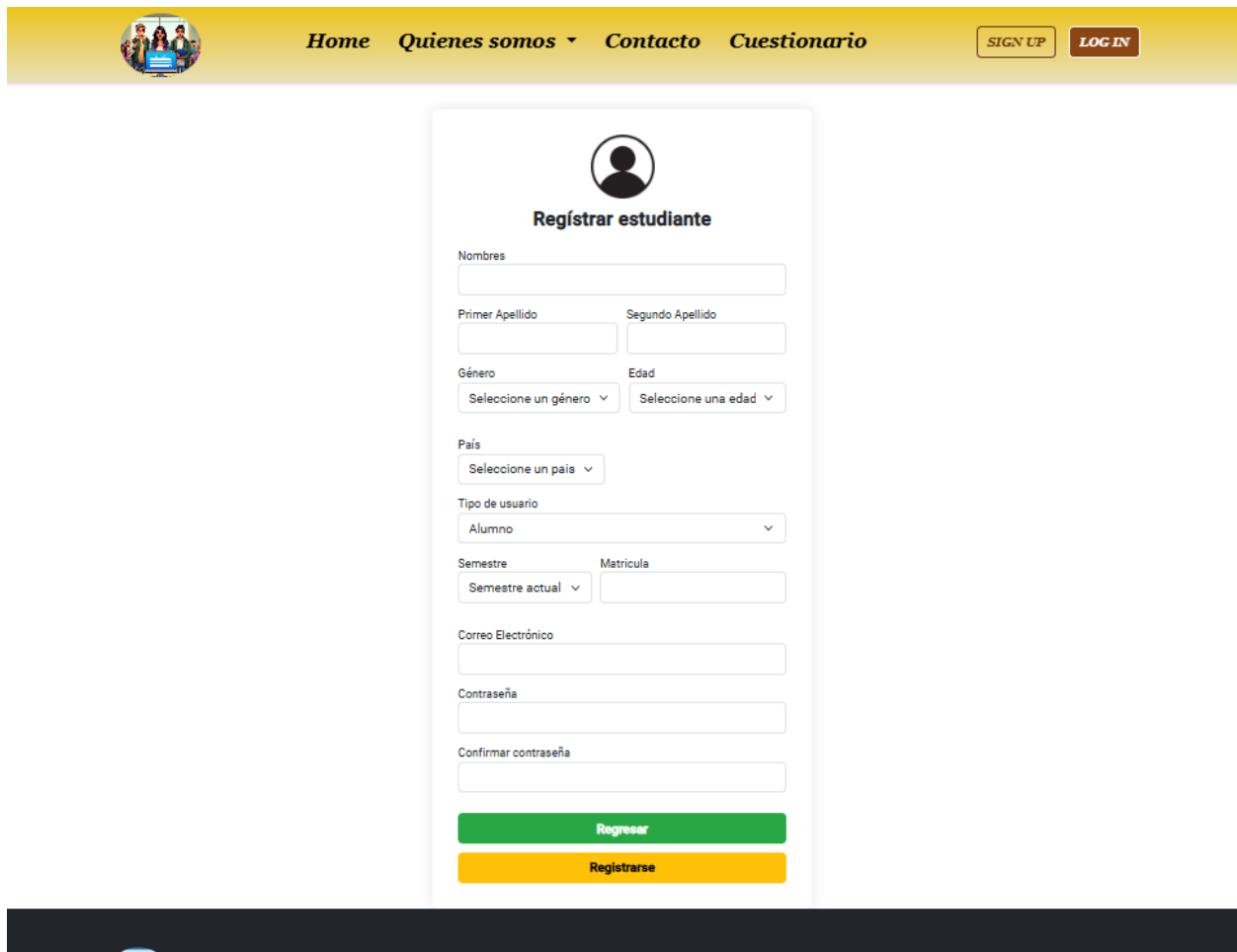
El Módulo de Registro permite a los nuevos usuarios crear una cuenta en la plataforma proporcionando información personal y académica detallada. Esta sección garantiza que solo los usuarios registrados puedan acceder a las funcionalidades avanzadas, como el cuestionario.

Al acceder al formulario de registro, el usuario verá los siguientes campos obligatorios:

- **Nombre:** Primer nombre del usuario.
- **Primer Apellido:** Primer apellido del usuario.
- **Segundo Apellido:** Segundo apellido del usuario.
- **Matrícula:** Número de identificación institucional para usuarios vinculados a universidades.
- **Semestre:** Semestre académico actual del usuario.
- **Institución:** Campo desplegable para seleccionar la institución a la que el usuario pertenece, basado en las instituciones participantes del proyecto.
- **País:** Selección del país de origen o residencia del usuario para una referencia geográfica.
- **Carrera:** Campo desplegable para seleccionar la carrera o programa académico del usuario.
- **Edad:** Campo numérico para ingresar la edad del usuario.
- **Género:** Selección de género con opciones estándar (e.g., femenino, masculino, otro).

- **Correo Electrónico:** Dirección de correo electrónico que servirá como nombre de usuario para el inicio de sesión y la recuperación de cuenta.
- **Contraseña:** Campo protegido para ingresar la contraseña, la cual debe cumplir con los requisitos de seguridad mínimos (longitud mínima, combinación de caracteres).
- **Confirmar Contraseña:** Campo adicional para volver a ingresar la contraseña, asegurando que coincide con la original.

Cada campo cuenta con validación de datos en tiempo real para asegurar precisión y cumplimiento con los requisitos de formato. Los campos obligatorios deben completarse adecuadamente antes de que el usuario pueda enviar el formulario. Al hacer clic en el botón "Registrarse", el sistema procesará y almacenará la información. Si el registro es exitoso, el usuario recibirá una confirmación y será redirigido a la pantalla de inicio de sesión para ingresar con sus nuevas credenciales.



The image shows a web application header with a logo and navigation links: Home, Quienes somos, Contacto, and Cuestionario. There are also buttons for SIGN UP and LOGIN. Below the header is a registration form titled "Registrar estudiante". The form contains the following fields:

- Nombres: A single text input field.
- Primer Apellido: A single text input field.
- Segundo Apellido: A single text input field.
- Género: A dropdown menu with the option "Seleccione un género".
- Edad: A dropdown menu with the option "Seleccione una edad".
- País: A dropdown menu with the option "Seleccione un país".
- Tipo de usuario: A dropdown menu with the option "Alumno".
- Semestre: A dropdown menu with the option "Semestre actual".
- Matricula: A single text input field.
- Correo Electrónico: A single text input field.
- Contraseña: A single text input field.
- Confirmar contraseña: A single text input field.

At the bottom of the form are two buttons: a green "Regresar" button and a yellow "Registrarse" button.

Ilustración 3. 11 Diseño del registro de usuario

MÓDULO DE INICIO DE SESIÓN

El Módulo de Inicio de Sesión permite a los usuarios registrados acceder a la plataforma. Incluye las siguientes funcionalidades:

- **Acceso:** Permite ingresar usando correo electrónico o matrícula y contraseña.
- **Registro de nuevos usuarios:** Acceso directo al módulo de registro para crear una cuenta proporcionando información básica.
- **Recuperación de contraseña:** Permite a los usuarios restablecer su contraseña mediante un enlace enviado a su correo electrónico.

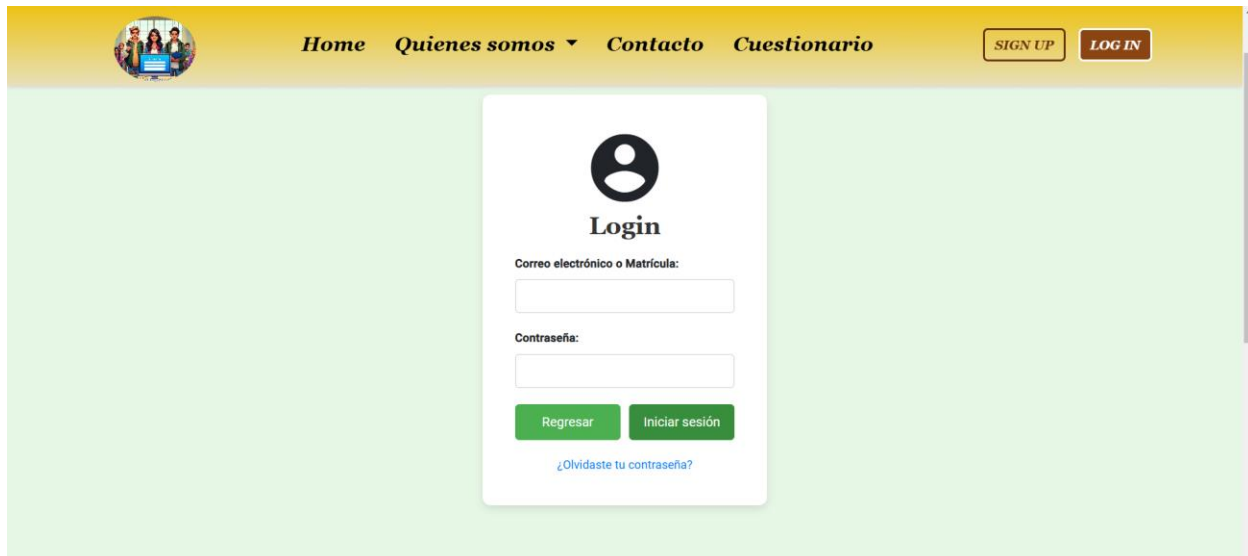


Ilustración 3. 12 Diseño del inicio de sesión

MÓDULO DE PERFIL DE USUARIO

Submódulo Perfil de Estudiante:

Este submódulo permite a los estudiantes administrar sus datos y acceder a los cuestionarios asignados por sus maestros. Incluye las siguientes funcionalidades:

- **Registro y Acceso (R11):** Los estudiantes pueden registrarse y acceder a la plataforma usando un nombre de usuario y contraseña únicos.
- **Visualización de cuestionarios (R12):** Los estudiantes pueden ver los cuestionarios que sus maestros les han asignado.
- **Respuesta a cuestionarios (R13):** Los estudiantes pueden responder a los cuestionarios desde cualquier dispositivo con acceso a internet.
- **Retroalimentación personalizada (R14):** Los estudiantes reciben comentarios específicos sobre sus características emprendedoras, basados en los resultados de los cuestionarios.
- **Seguimiento del progreso (R15):** Los estudiantes pueden ver su progreso a lo largo de la carrera, utilizando los resultados acumulados de los cuestionarios.

- **Consulta de datos de perfil (R16):** Los estudiantes pueden revisar y actualizar su información de perfil.



Ilustración 3. 13 Diseño del dashboard de estudiante

Submódulo Perfil de Maestro:

Este submódulo permite a los maestros gestionar los cuestionarios asignados a sus alumnos y acceder a los resultados individuales y grupales. Incluye las siguientes funcionalidades:

- **Asignación de cuestionarios (R17):** Los maestros pueden asignar cuestionarios a grupos específicos, utilizando filtros como carrera, grado o sección.
- **Configuración de fechas (R18):** Los maestros pueden establecer fechas de inicio y fin para cada cuestionario asignado.
- **Visualización de respuestas (R19):** Los maestros pueden ver las respuestas de los alumnos, tanto de manera individual como grupal.
- **Generación de informes (R20):** Los maestros pueden generar informes detallados de los resultados, con información sobre cualidades y áreas de mejora de cada estudiante, y recibir retroalimentación dividida por sección.
- **Exportación de informes (R21):** Los informes pueden exportarse en formatos PDF o XML.
- **Generación de códigos de acceso (R22):** Los maestros pueden crear códigos de acceso únicos para cada grupo, para controlar el acceso a los cuestionarios.
- **Consulta de datos de perfil (R23):** Los maestros pueden revisar y actualizar su perfil.

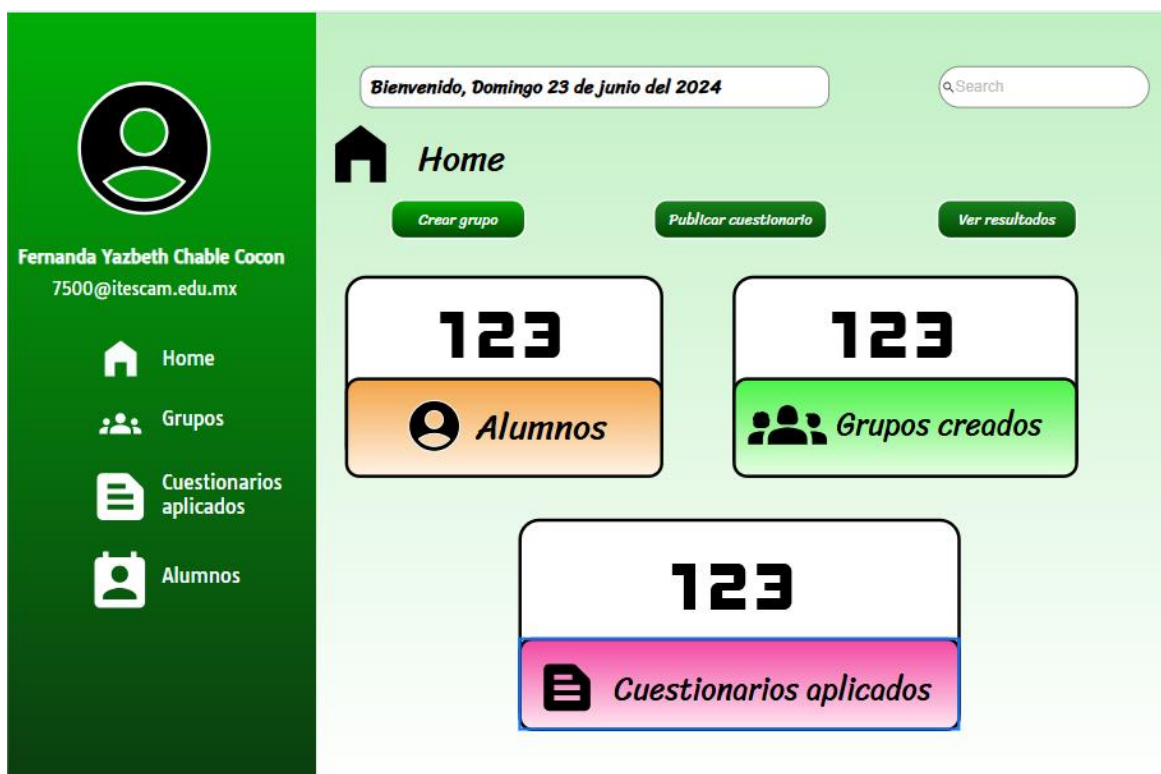


Ilustración 3. 14 Diseño del perfil del maestro

Submódulo Perfil Administrativo:

Este submódulo proporciona a los administrativos acceso completo para gestionar usuarios y datos de la plataforma, así como generar informes detallados y realizar un seguimiento del uso de la plataforma. Incluye las siguientes funcionalidades:

- **Gestión de usuarios (R24):** Los administrativos pueden crear, editar y eliminar usuarios, incluyendo estudiantes, maestros y otros administradores.
- **Gestión de categorías, expresiones y variables (R25):** Los administrativos pueden administrar las categorías, expresiones y variables relacionadas con las preguntas de los cuestionarios.
- **Visualización de resultados (R26):** Los administrativos pueden ver los resultados de todas las encuestas realizadas en la plataforma.
- **Generación de informes detallados (R27):** Los administrativos pueden generar informes con gráficos y tablas, mostrando cualidades individuales y grupales, segmentadas por carrera, semestre y generación.
- **Exportación de informes (R28):** Los informes generados pueden exportarse en formatos PDF o XML.
- **Consulta de datos de perfil (R29):** Los administrativos pueden consultar los datos de perfil de todos los usuarios en la plataforma.

- **Seguimiento de uso (R30):** Los administrativos pueden realizar un seguimiento del uso y actividad en la plataforma.
- **Retroalimentación personalizada (R31):** Los administrativos pueden brindar retroalimentación específica a los alumnos según sus respuestas en los cuestionarios.
- **Almacenamiento de respuestas y retroalimentación (R32):** Las respuestas y retroalimentación de los cuestionarios se almacenan para realizar un seguimiento del progreso individual y grupal a lo largo de la carrera.

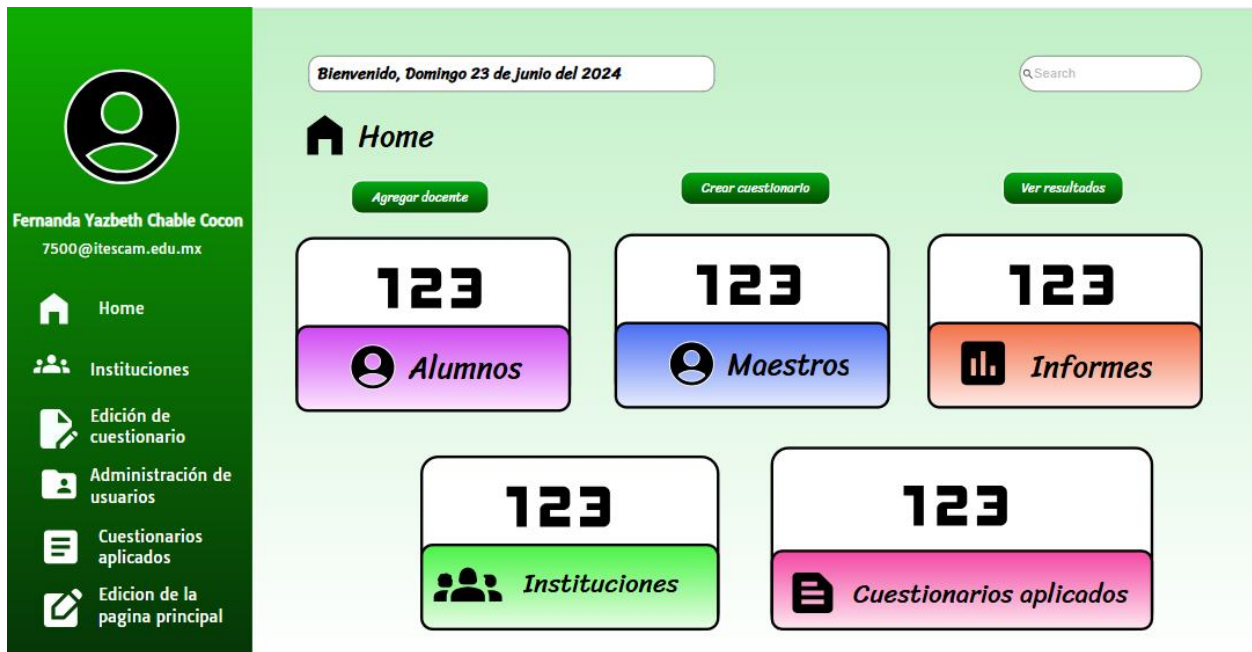


Ilustración 3. 15 Diseño del perfil administrativo

MÓDULO DE GESTIÓN DE CUESTIONARIOS

Este módulo permite a los administradores y maestros gestionar todos los aspectos relacionados con los cuestionarios asignados a los estudiantes, incluyendo la edición de preguntas, asignación de cuestionarios, retroalimentación y el cálculo de resultados.

Submódulo Edición de Cuestionarios:

Este submódulo permite a los administradores editar y ajustar los contenidos de los cuestionarios de acuerdo con las necesidades de evaluación.

- **Edición de Preguntas, Variables, Expresiones y Categorías (R33):** Los administradores pueden modificar las preguntas, variables, expresiones y categorías de cada cuestionario y definir si cada variable tiene un impacto positivo o negativo en el resultado final. Esto permite ajustar los cuestionarios para mejorar la precisión en la evaluación de competencias.

Submódulo Configuración del Cuestionario:

Este submódulo permite a los administradores y maestros establecer los parámetros específicos de cada cuestionario asignado a un grupo.

- **Configuración de cuestionarios para grupos (R34):** Permite definir un cuestionario específico para un grupo y establecer una fecha límite para su finalización, asegurando que todos los estudiantes lo completen dentro del tiempo establecido.

<i>Cuestionario</i>	<i>Variable</i>	<i>Signo</i>
Pregunta 1	Select	Select
Pregunta 1	Select	Select
Pregunta 1	Select	Select
Pregunta 1	Select	Select
Pregunta 1	Select	Select
Pregunta 1	Select	Select
Pregunta 1	Select	Select

Anterior **Siguiente** **Cancelar** **Guardar**

Ilustración 3. 16 Diseño preliminar de la edición del cuestionario

Submódulo Asignación de Cuestionarios:

Este submódulo facilita a los maestros la asignación de cuestionarios a grupos completos.

- **Asignación de cuestionarios (R35):** Los maestros pueden asignar cuestionarios a grupos completos, controlando así el acceso al contenido evaluativo.



Ilustración 3. 17 Mensaje de activación del cuestionario

Submódulo Código de Acceso:

Este submódulo asegura que solo los estudiantes autorizados puedan acceder al cuestionario.

- **Generación de código de acceso (R36):** Los estudiantes reciben un código de acceso exclusivo que permite el ingreso al cuestionario asignado, restringiendo el acceso solo a los estudiantes autorizados.

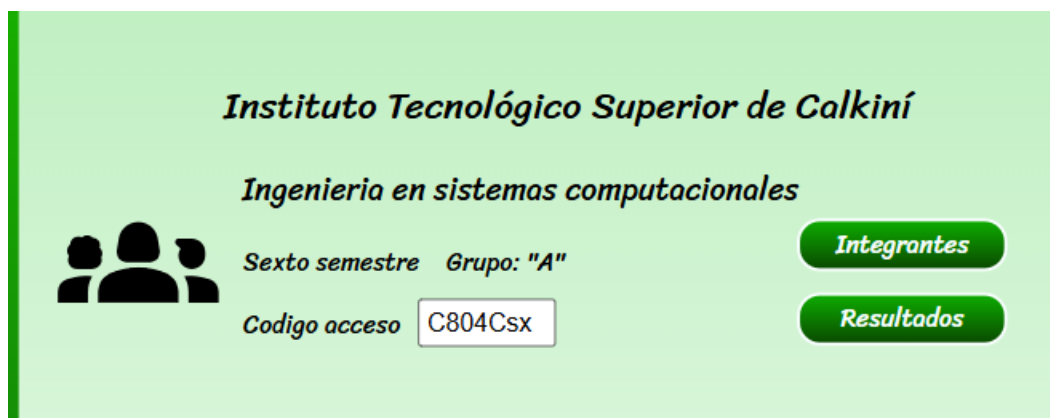


Ilustración 3. 18 Diseño de la información del grupo (código de acceso)

Submódulo Respuestas de los Estudiantes:

Este submódulo permite a los estudiantes responder a los cuestionarios de forma sencilla y accesible.

- **Respuestas en línea (R37):** Los estudiantes pueden completar y enviar sus respuestas a los cuestionarios en línea, independientemente del dispositivo que estén utilizando.

Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs)²

INSTRUCCIONES

a) El cuestionario consta de 55 declaraciones breves. Lea cuidadosamente cada una y decida qué tan acertadamente le describe. Sea honesto consigo mismo. Recuerde que nadie tiene puntaje perfecto. Este es un autoanálisis, por lo tanto el único que juzgará su resultado es usted mismo

b) Seleccione el número que corresponde para designar la media en que cada declaración describe su comportamiento:

1= Nunca	2= Raras veces	3 = Algunas veces	4= Usualmente	5= Siempre
----------	----------------	-------------------	---------------	------------

c) Algunas declaraciones pueden ser similares, pero ninguna es igual, por lo tanto deben ser respondidas todas

d) Si tiene duda sobre alguna, pase a la siguiente, y al finalizar vuelva sobre las que dejó sin contestar

Cancelar

Empezar

Ilustración 3. 19 Diseño de la vista de las instrucciones del cuestionario

Cuestionario Estándar de Autoevaluación de las
 Características Emprendedoras Personales (CEPs)²

Lea cuidadosamente cada una y decida qué tan acertadamente le describe. Sea honesto consigo mismo.

- 1) *Me esmero en buscar cosas que necesitan hacerse.*
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
- 2) *Cuando me enfrento a un problema difícil Invierto gran cantidad de tiempo en encontrar una solución.*
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
- 3) *Termino mi trabajo a tiempo.*
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
- 4) *Me molesta cuando las cosas no se hacen debidamente.*
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
- 5) *Prefiero situaciones en las que puedo controlar la mayor cantidad de la información al máximo.*
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
- 6) *Me gusta pensar sobre el futuro.*
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
- 7) *Cuando comienza una tarea o un proyecto nuevo, reúno toda la información posible antes de darle curso.*
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
- 8) *Planifico un proyecto grande dividiéndolo en tareas de menor tamaño*
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
- 9) *Logro que otros apoyen mis recomendaciones.*
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
- 10) *Me siento con fiado en que puede tener éxito en cualquier actividad que me propongo ejecutar.*
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre

Anterior
Siguiente

Ilustración 3. 20 Diseño del cuestionario

Submódulo Retroalimentación Inmediata:

Este submódulo proporciona a los estudiantes una respuesta rápida después de enviar sus cuestionarios.

- **Retroalimentación inmediata (R38):** Tras enviar sus respuestas, los estudiantes reciben retroalimentación instantánea sobre sus características personales, destacando sus fortalezas y áreas de mejora, además de recibir recomendaciones.

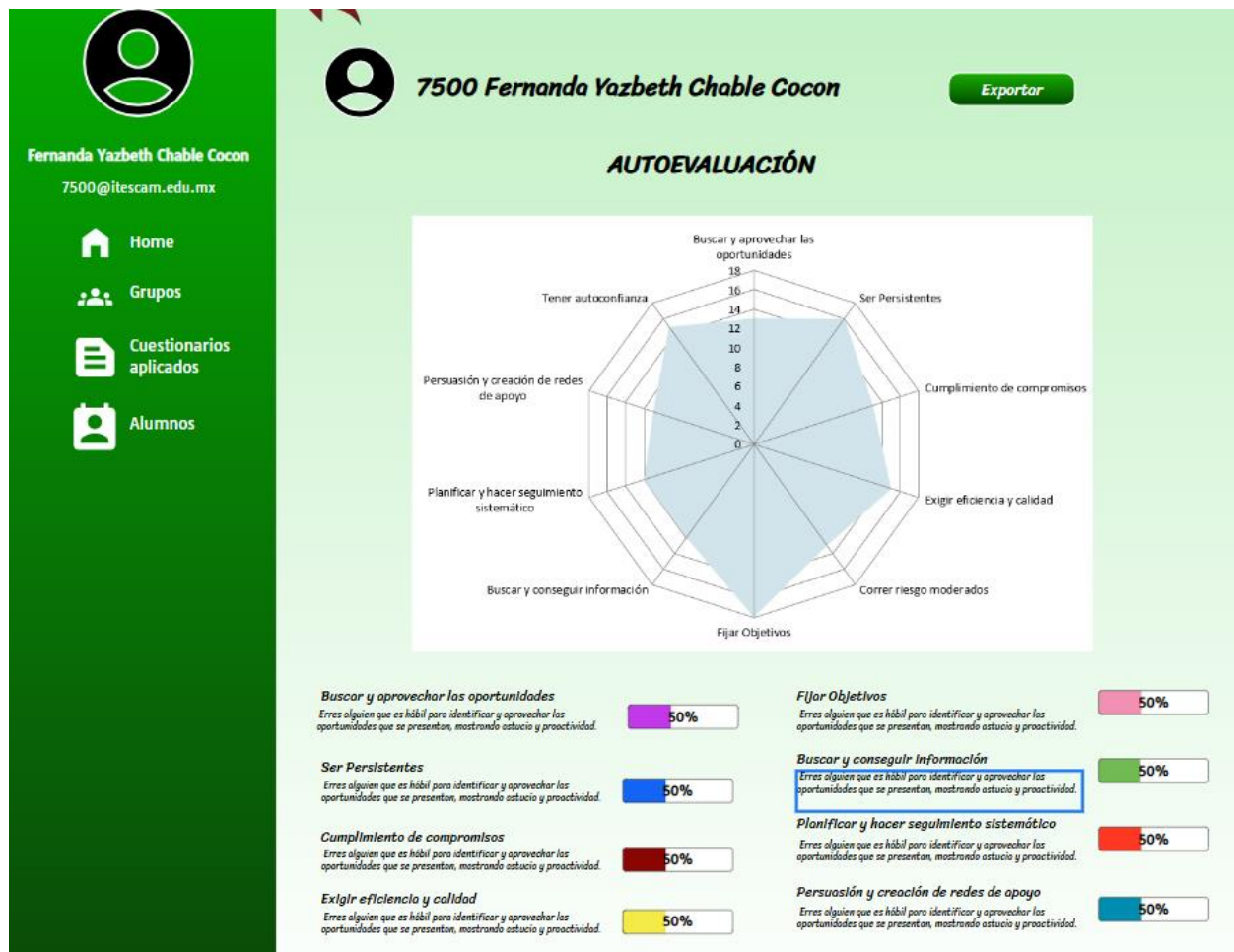


Ilustración 3. 21 Diseño de resultados del cuestionario

Submódulo Eliminación de Cuestionarios Completados:

Este submódulo optimiza la interfaz del estudiante al ocultar los cuestionarios ya completados.

- **Eliminación de cuestionarios completados (R39):** Una vez que el estudiante ha enviado el cuestionario, este desaparece de su lista para evitar confusión y mantener su pantalla de inicio organizada.

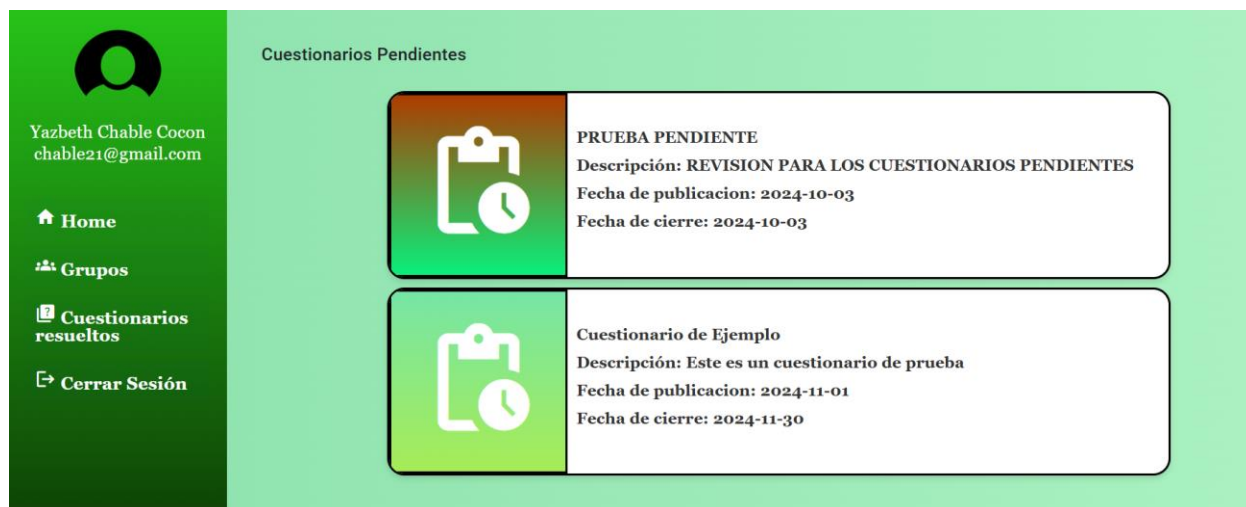


Ilustración 3. 22 Diseño de vista de los cuestionarios pendientes

MÓDULO DE GESTIÓN DE GRUPOS

Este módulo permite a los maestros gestionar los grupos de estudiantes, desde la creación de grupos hasta la asignación de los estudiantes a estos grupos, garantizando que la organización sea eficiente y adecuada.



Ilustración 3. 23 Diseño de módulo de gestión de grupos

Submódulo Creación de Grupos:

Este submódulo permite a los maestros crear y gestionar grupos de estudiantes de acuerdo con parámetros específicos como la carrera y el grado de los estudiantes.

- **Creación y gestión de grupos (R43):** Los maestros pueden formar grupos de estudiantes en función de su carrera, generación, semestre y nombre. Este submódulo permite crear grupos específicos y asignarles un código de acceso único, asegurando que cada grupo tenga la estructura adecuada para su participación en los cuestionarios y actividades.

Yazbeth Cocon Chable
yazbethcocon@gmail.com

Home

Grupos

Cuestionarios aplicados

Alumnos

Cerrar Sesión

Crear grupo

Nombre*

Semestre*

Selecciona una Carrera*

Elige una carrera

Selecciona una Generación*

Elige una generación

Cancelar

Crear Grupo

Ilustración 3. 24 Diseño del módulo para la creación de los grupos

Yazbeth Cocon Chable
yazbethcocon@gmail.com

Home

Grupos

Cuestionarios aplicados

Alumnos

Cerrar Sesión

Crear grupo

¡Grupo Creado Exitosamente!

El código de acceso es: QjoWWN1h

Aceptar

Cancelar

Crear Grupo

Ilustración 3. 25 Mensaje de grupo creado exitosamente

Submódulo Asignación a Grupos:

Este submódulo facilita que los estudiantes se asignen a grupos específicos, utilizando códigos de acceso proporcionados por los maestros.

- **Asignación de estudiantes a grupos (R44):** Los estudiantes pueden unirse a un grupo al ingresar un código de acceso único proporcionado por el maestro. Esto asegura que solo los

estudiantes autorizados se asignen a los grupos correspondientes, manteniendo la organización y control dentro de la plataforma.

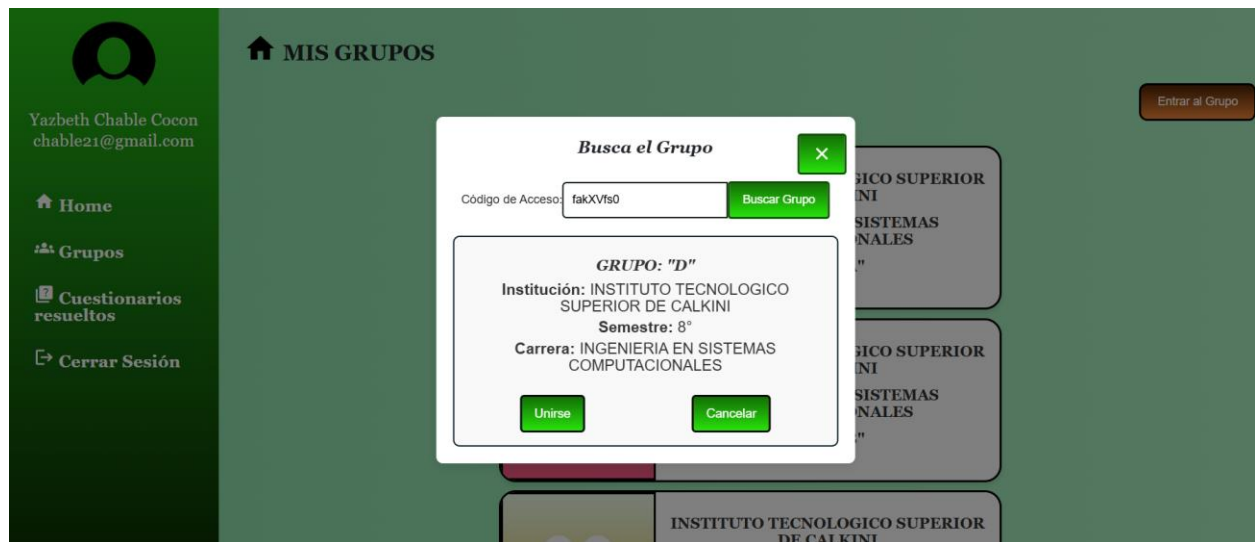


Ilustración 3. 26 Diseño de vista para la búsqueda de grupos

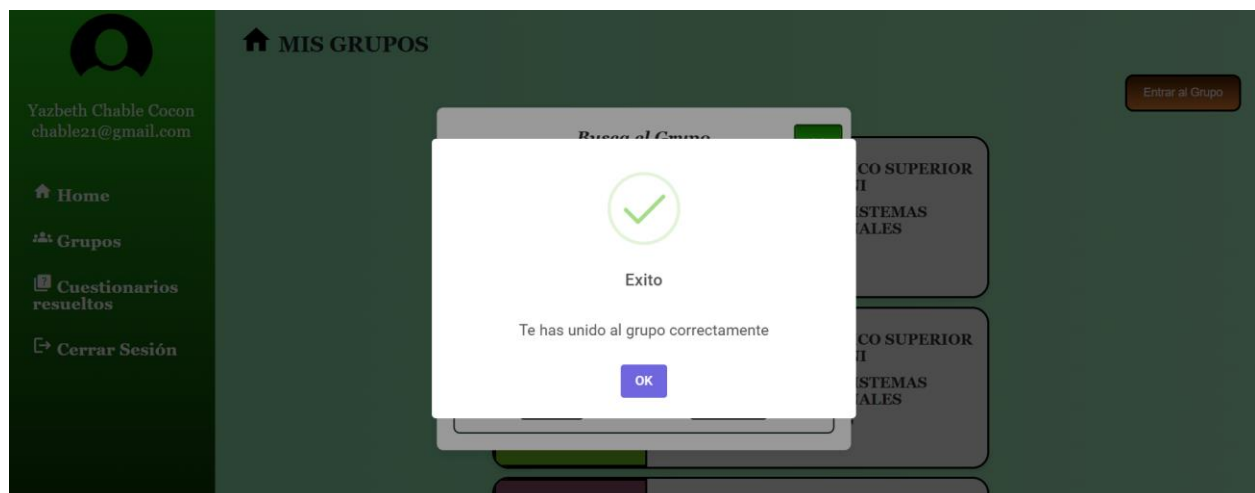


Ilustración 3. 27 Diseño de mensaje de éxito al unirse a un grupo

MÓDULO DASHBOARD

Este módulo proporciona una vista integral y personalizada según el tipo de usuario (Estudiante, Maestro, Administrativo), permitiendo visualizar información clave sobre los cuestionarios y su rendimiento dentro de la plataforma.

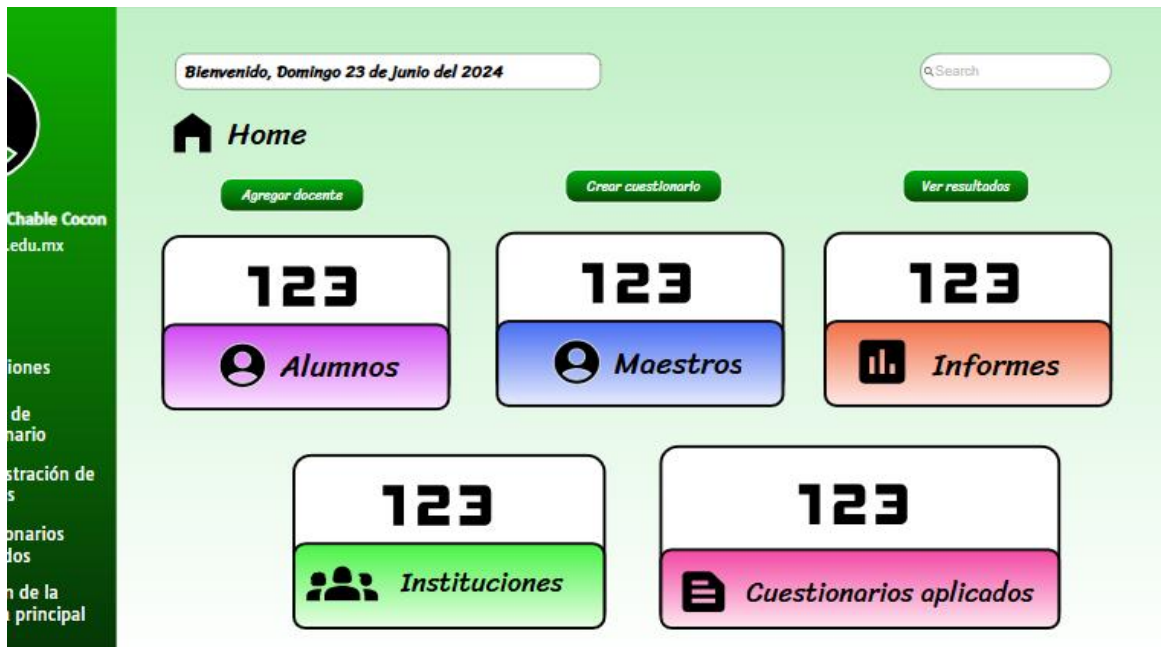


Ilustración 3. 28 Diseño de vista preliminar para el dashboard administrativo

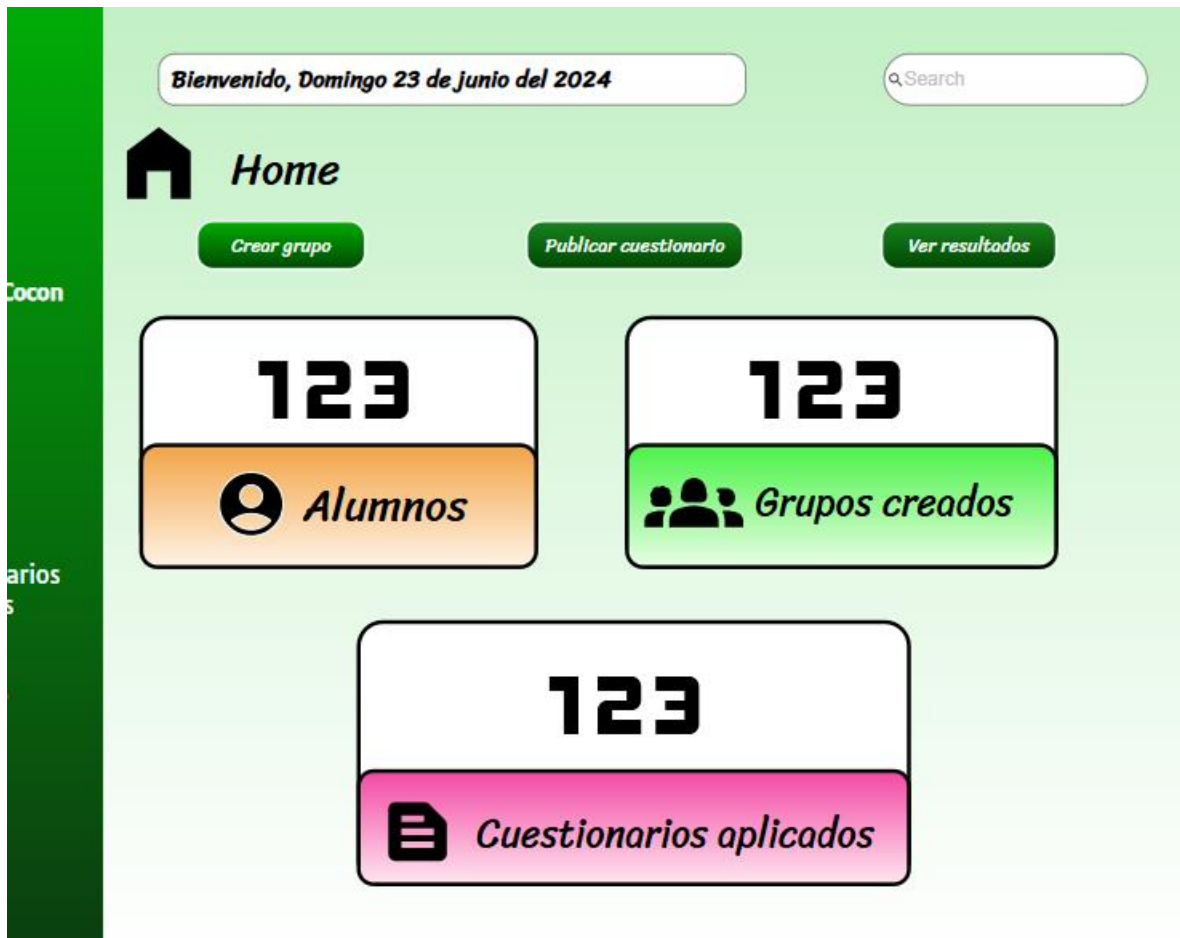


Ilustración 3. 29 Diseño de vista preliminar para el dashboard para maestros

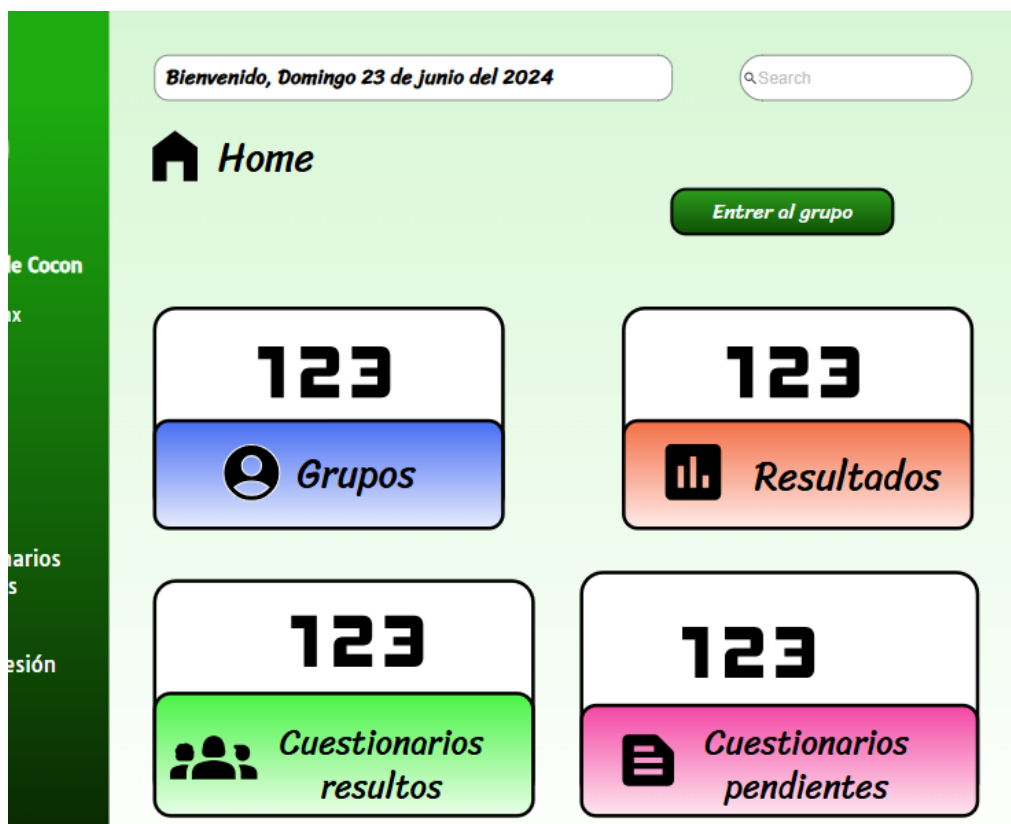


Ilustración 3. 30 Diseño de vista preliminar para el dashboard para alumnos

Submódulo Dashboard para Estudiantes:

Este submódulo ofrece a los estudiantes una vista general de su progreso académico y de los cuestionarios completados.

- **Visualización de resultados y progreso (R45):** Los estudiantes pueden acceder a un resumen completo de los cuestionarios que han completado, mostrando tanto los resultados obtenidos como su avance en función de los cuestionarios realizados. Esto permite a los estudiantes llevar un seguimiento de su desempeño y entender su evolución académica.

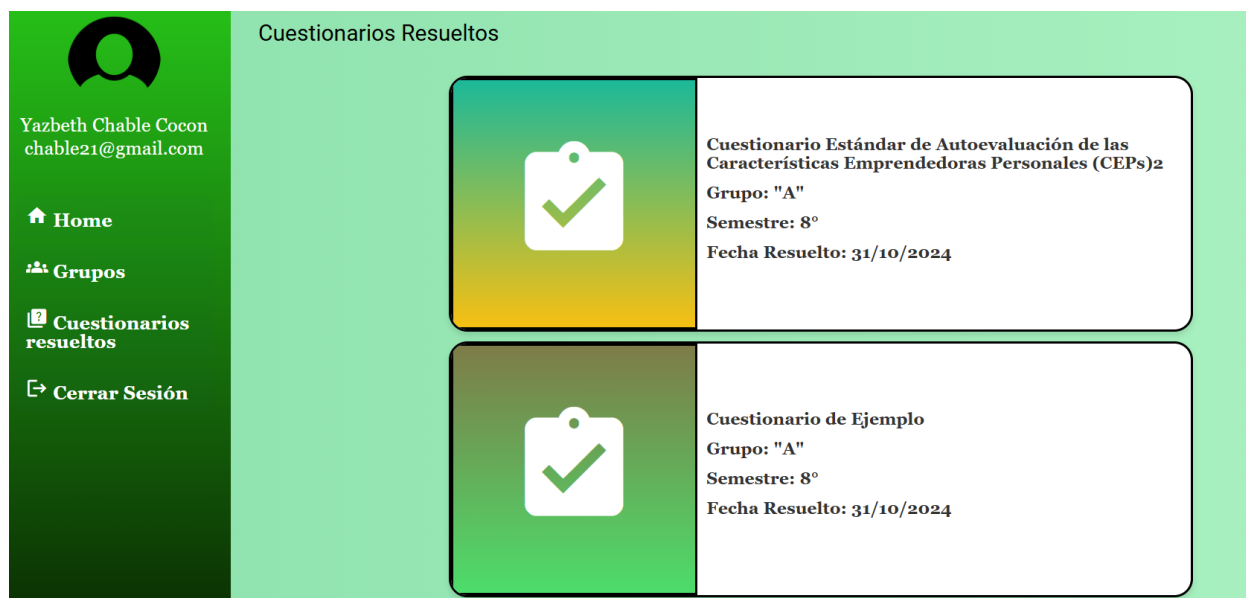


Ilustración 3. 31 Visualización de los cuestionarios resueltos (Alumno)

Submódulo Dashboard para Maestros:

Este submódulo está diseñado para ofrecer a los maestros una vista detallada de los cuestionarios y grupos que gestionan, permitiéndoles acceder a los resultados tanto individuales como grupales.

- **Listado de cuestionarios y resultados (R46):** Los maestros pueden ver todos los cuestionarios que han asignado, así como los grupos de estudiantes que gestionan. Además, podrán acceder a los resultados de los cuestionarios, tanto a nivel individual como grupal, lo que les permite monitorear el rendimiento de sus estudiantes de manera eficiente.



Ilustración 3. 32 Diseño de la vista de los grupos



Ilustración 3. 33 Diseño de la vista para la gestión de alumnos de un grupo

Submódulo Dashboard para Administrativos:

Este submódulo proporciona a los administrativos una vista completa sobre el uso y desempeño general de la plataforma.

- **Acceso a resultados y administración (R47):** Los administrativos tienen acceso a todos los resultados de los cuestionarios aplicados, así como estadísticas detalladas sobre los estudiantes y los grupos. También podrán consultar información relacionada con la gestión de usuarios y otros aspectos administrativos de la plataforma.

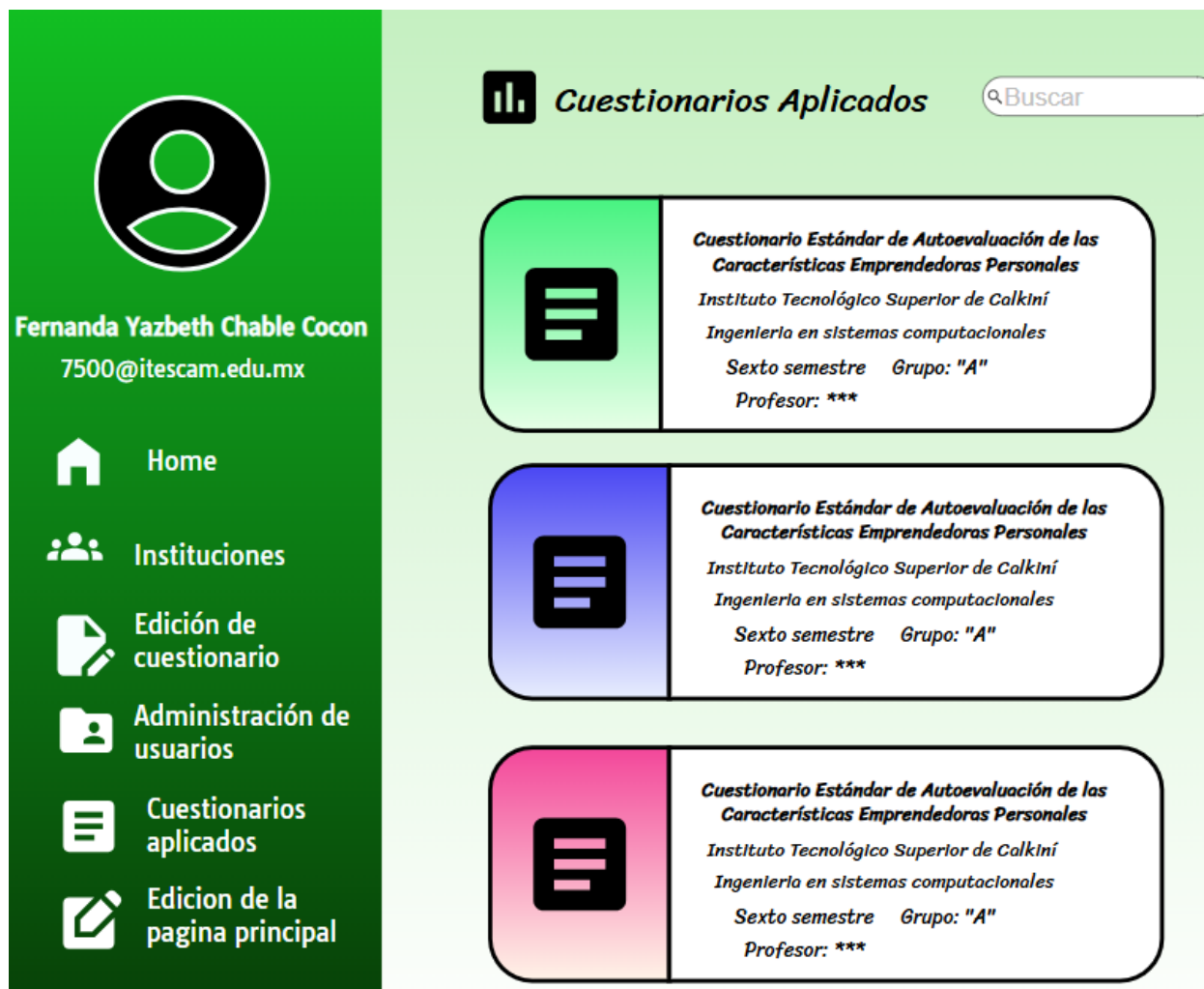


Ilustración 3. 34 Diseño de cuestionarios aplicados para el administrador



Ilustración 3. 35 Diseño preliminar para la administración de usuarios (Administradores)



Ilustración 3. 36 Diseño preliminar para la lista de usuarios (Dashboard de administradores)

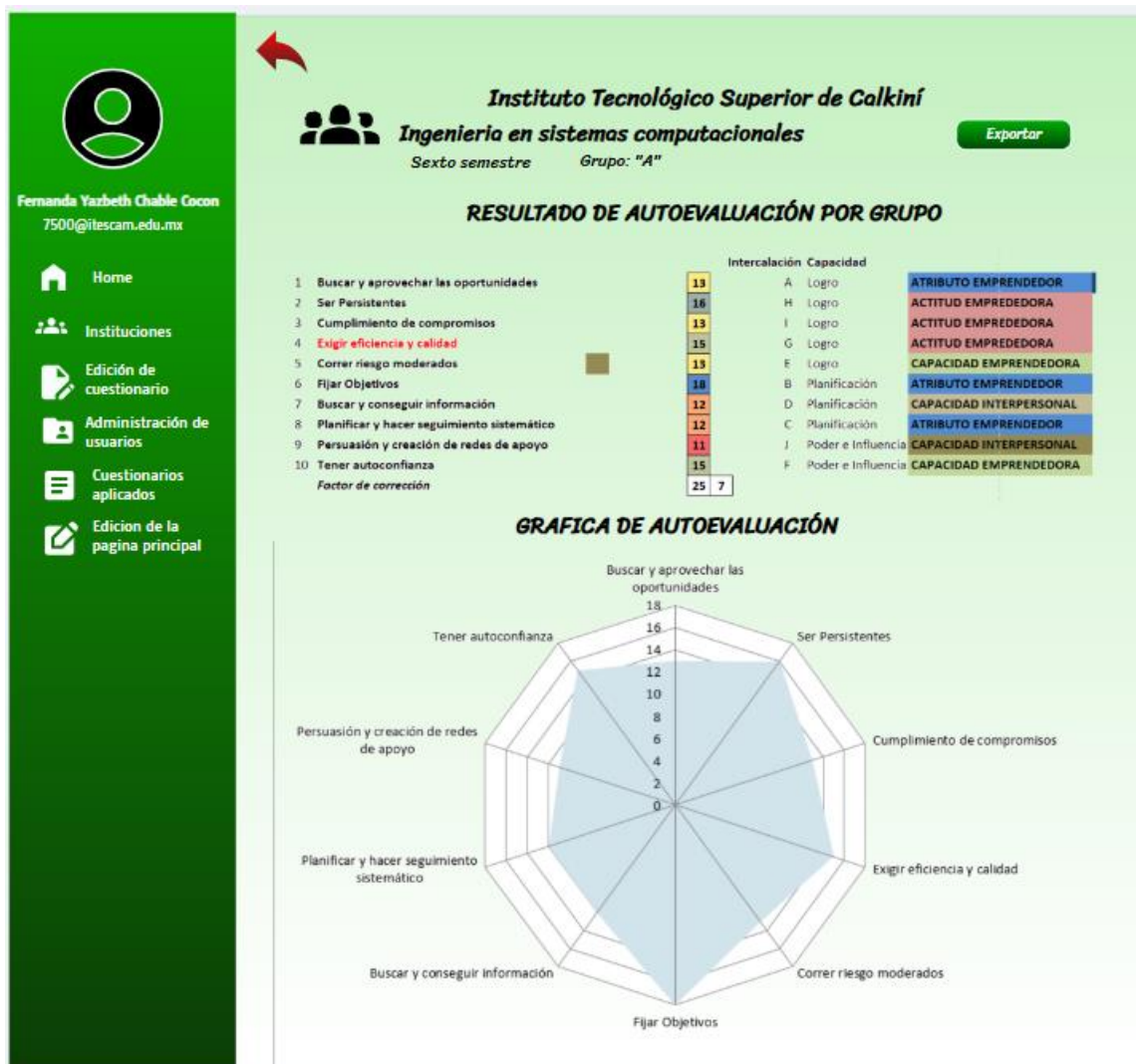


Ilustración 3. 37 Diseño preliminar de la vista para resultados

MÓDULO DE ANÁLISIS Y REPORTE

Este módulo tiene como objetivo proporcionar análisis detallados sobre los resultados de los cuestionarios, visualizaciones gráficas y la generación de informes para los administradores, permitiéndoles tomar decisiones informadas sobre el progreso de los estudiantes.

Submódulo Gráficas de Resultados:

Este submódulo permite a los usuarios visualizar los resultados de los cuestionarios de manera gráfica.

- Visualización de resultados (R48): Los resultados de los cuestionarios se presentan mediante gráficos, facilitando su análisis.

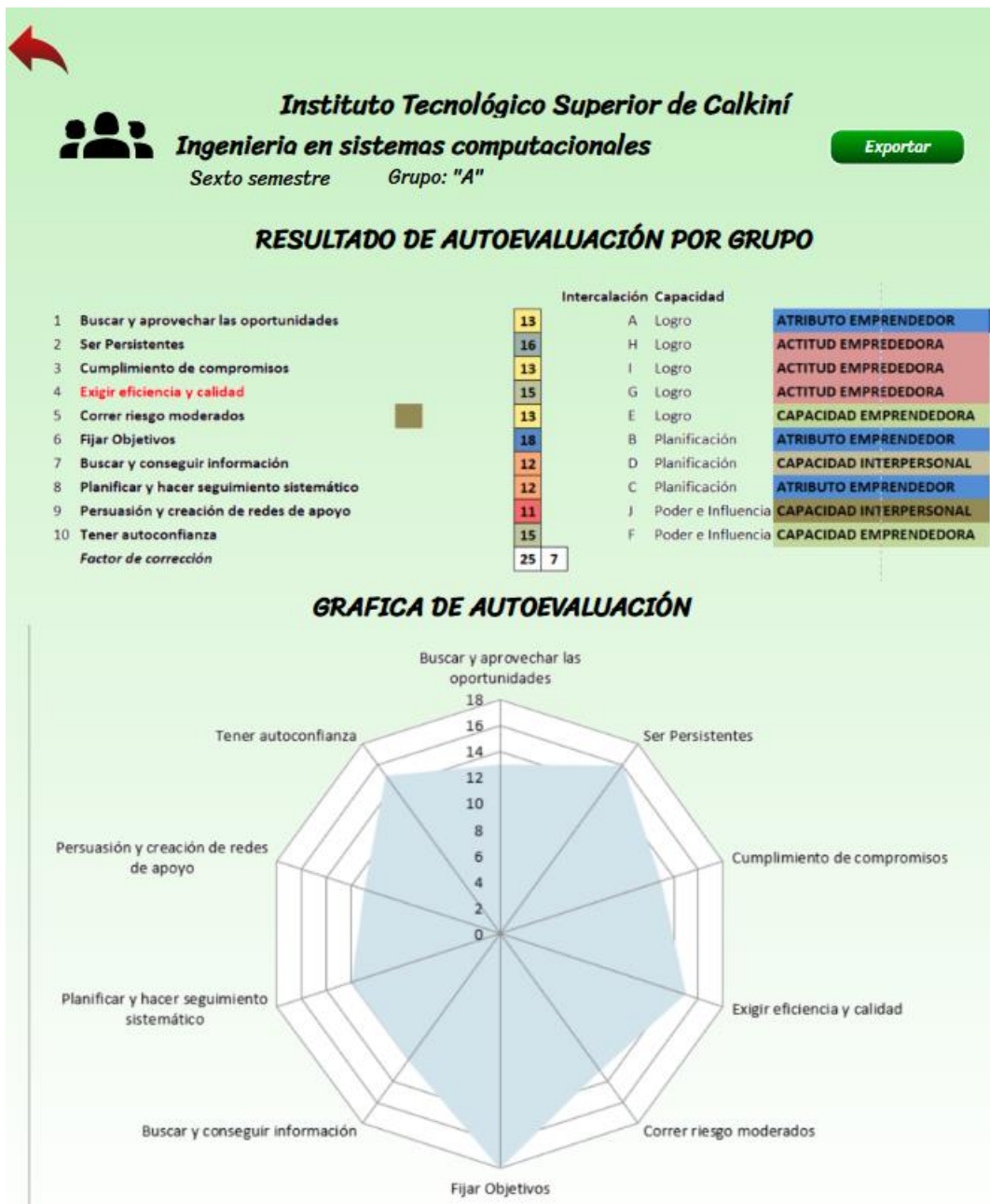


Ilustración 3. 38 Diseño preliminar para resultados

Submódulo Exportación a PDF y XML:

Este submódulo permite exportar los resultados de los cuestionarios en formato PDF.

- **Exportación de resultados en PDF y XML (R52):** Las gráficas de resultados pueden ser exportadas en formato PDF, ya sea a nivel individual por estudiante o grupal. Esto permite a los administradores o maestros generar informes visuales fáciles de compartir y analizar.

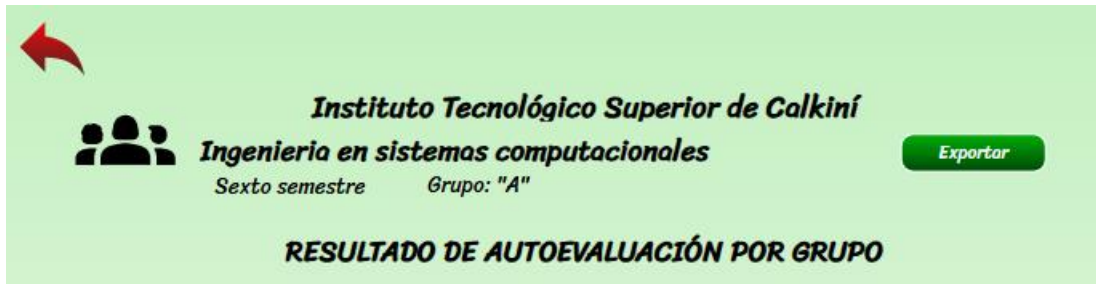


Ilustración 3. 39 Diseño preliminar para exportaciones de resultados grupales

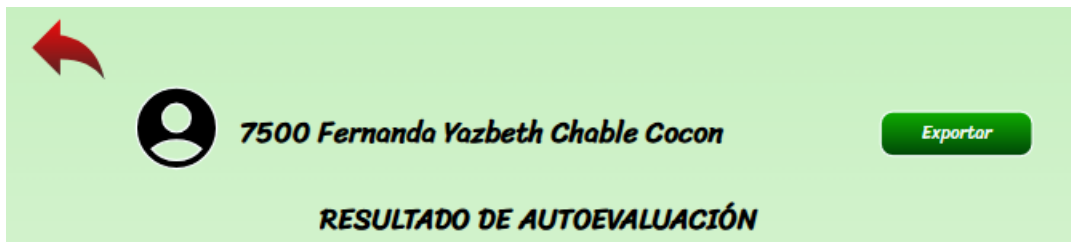


Ilustración 3. 40 Diseño preliminar para exportaciones de resultados individuales

3.6. CASOS DE USO

3.6.1 DIAGRAMA CASO DE USO PARA INICIAR SESION

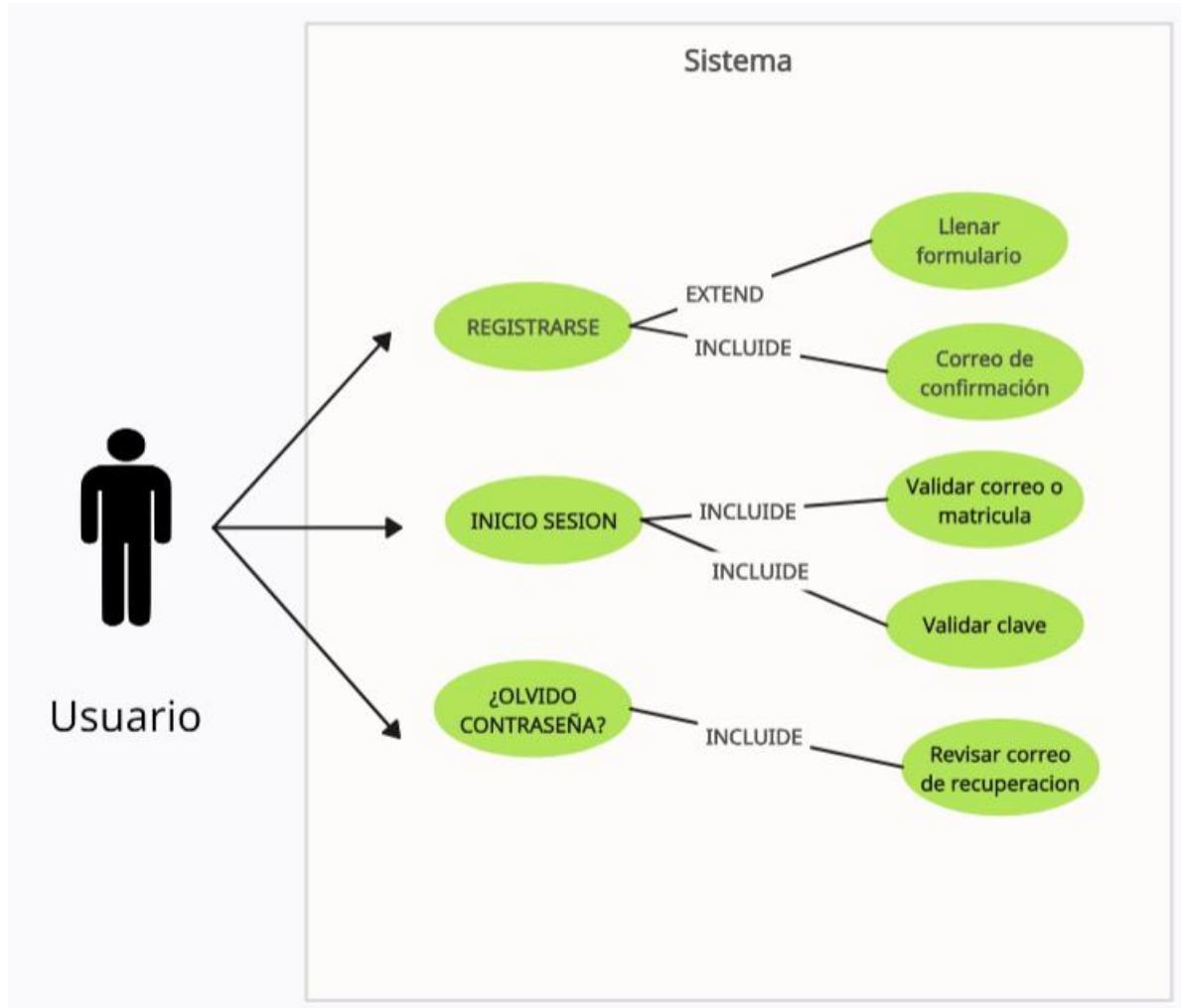


Ilustración 3. 41 Diagrama de uso para inicio sesión

3.6.2 DIAGRAMA CASO DE USO PARA EL ESTUDIANTE

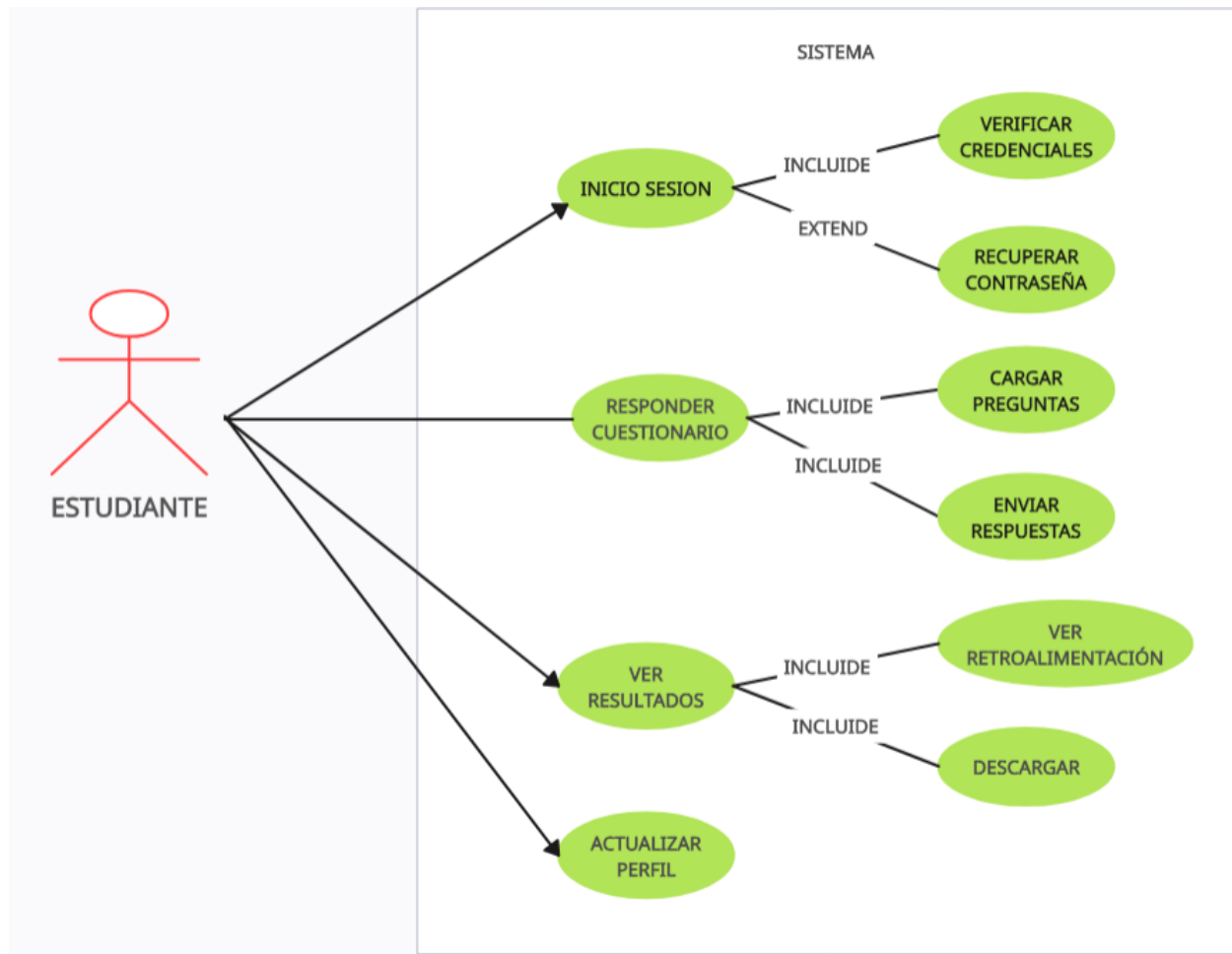


Ilustración 3. 42 Diagrama de uso para el estudiante

3.6.3 DIAGRAMA CASO DE USO PARA EL MAESTRO

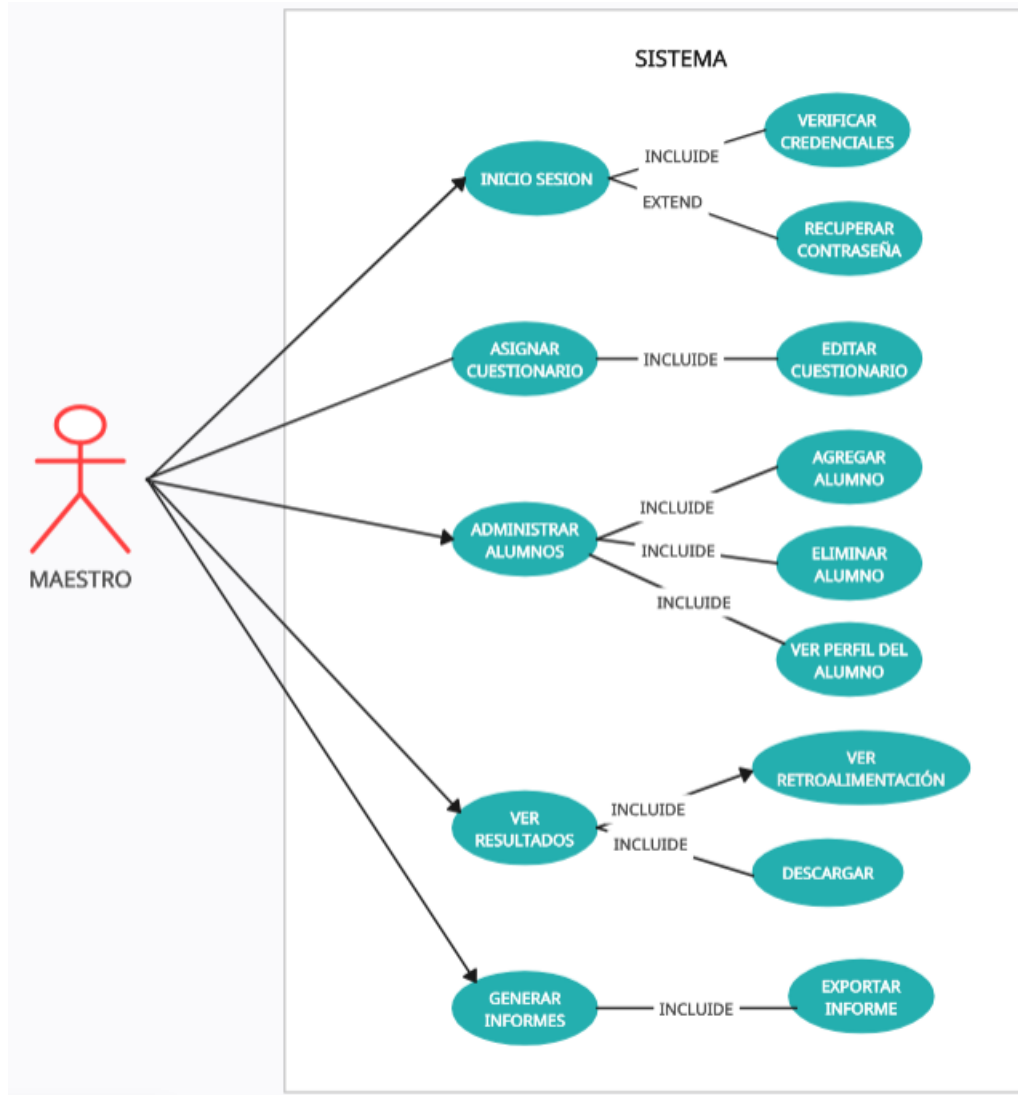


Ilustración 3. 43 Diagrama de uso para el maestro

3.6.4 DIAGRAMA CASO DE USO PARA EL ADMINISTRADOR

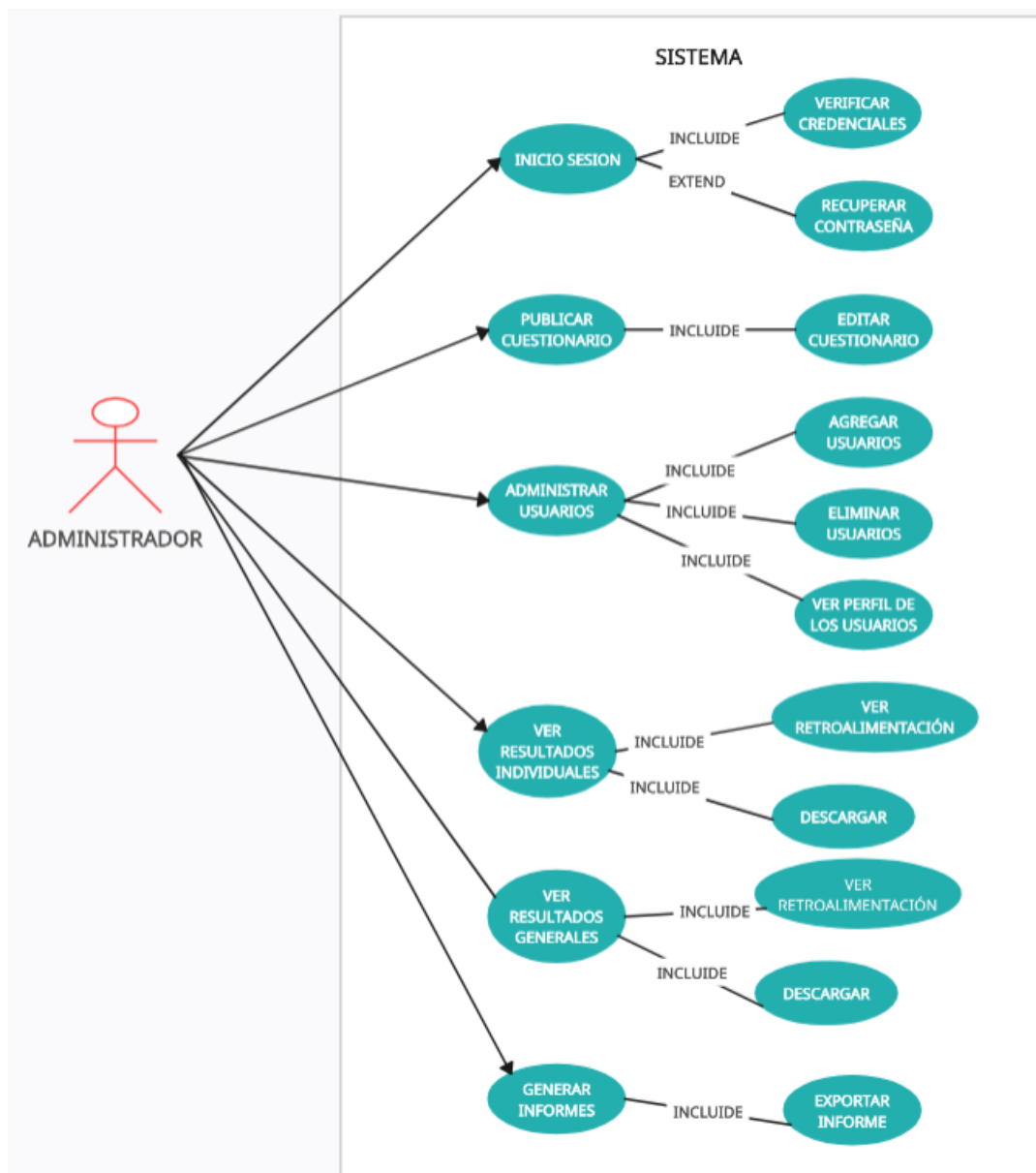


Ilustración 3. 44 Diagrama de uso para el administrador

3.7. MODELO RELACIONAL

Después del levantamiento de los requerimientos del sistema se diseñó el modelo ER para la base de datos que nos servirá para guardar todos los datos del sistema, tanto de los alumnos, maestros, administradores y el cuestionario.

Es importante destacar que la base de datos esta desarrollada en MySQL, ya que es una de las herramientas que se utilizaran por su fácil manejo. De igual manera para la gestión de la base de datos se han utilizado (Apis de servicio) por la comunidad y el fácil manejo de ellas.

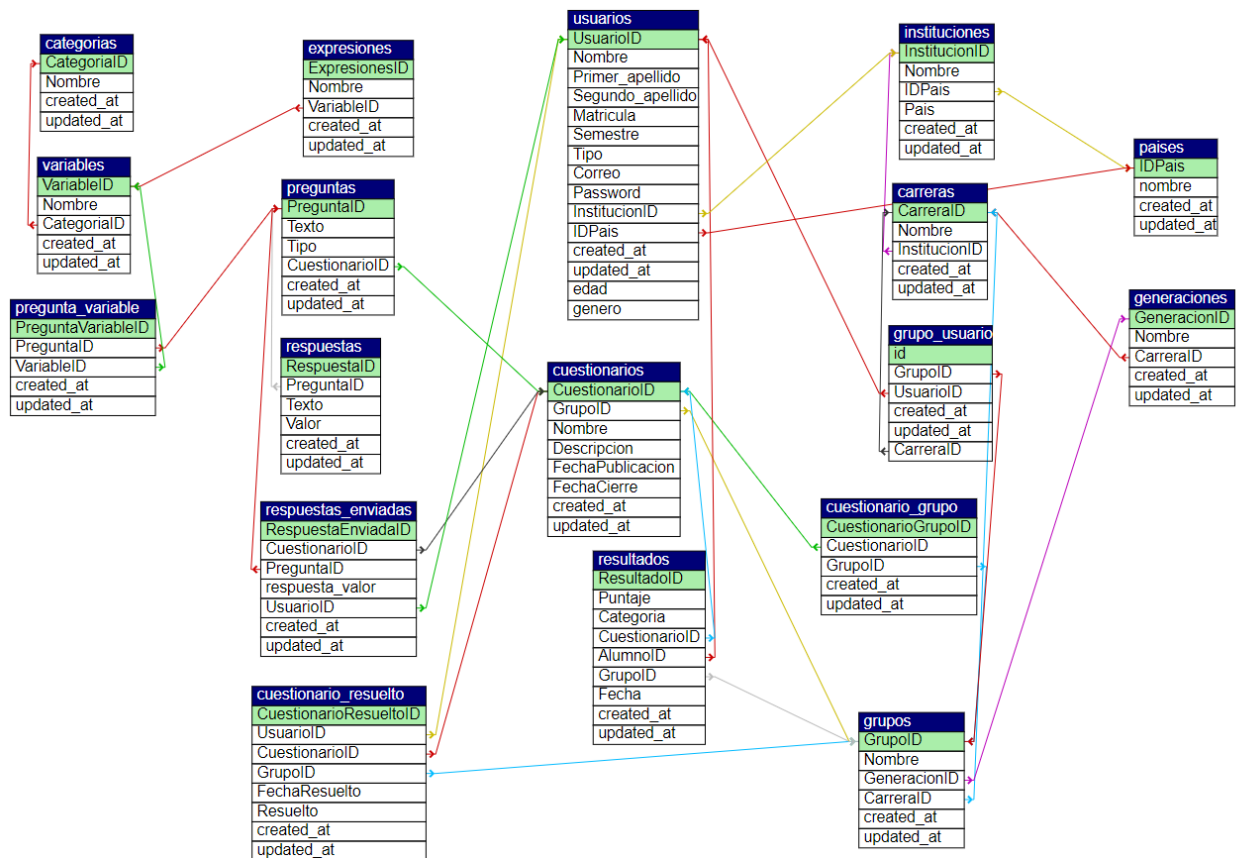


Ilustración 3. 45 Modelo Relacional

3.8. IMPLEMENTACIÓN Y FUNCIONAMIENTO DEL SISTEMA DE INTELIGENCIA DE NEGOCIOS.

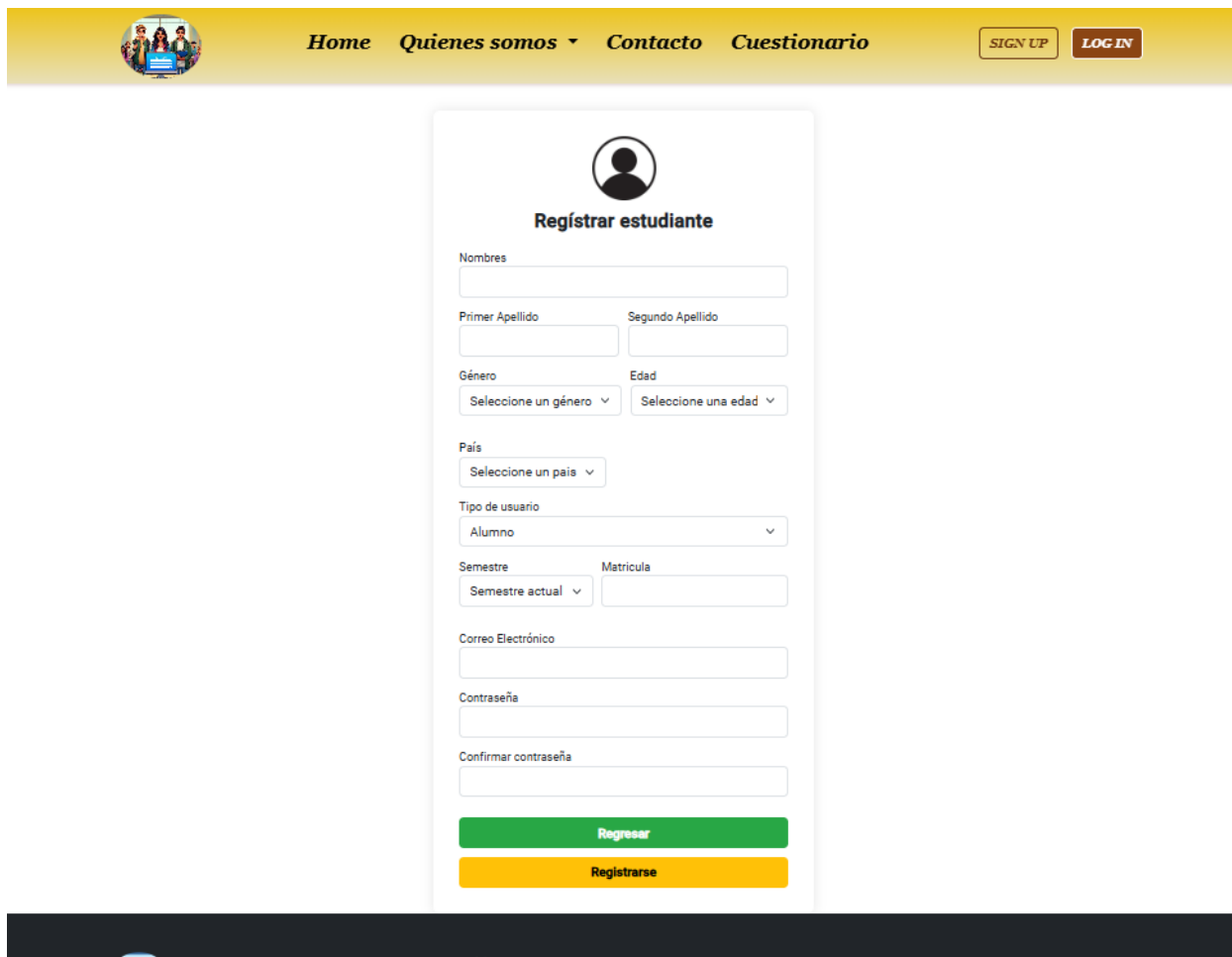
En esta sección se describe el desarrollo e implementación del sistema de inteligencia de negocios orientado a la identificación y mejora de perfiles emprendedores. Se detallan las principales funcionalidades, la estructura de la plataforma.

El sistema cuenta con una interfaz intuitiva que permite a los usuarios acceder a diversas secciones desde un menú principal, entre ellas: Home, Quienes somos, Contacto, Cuestionario, Registro e inicio de sesión. La pantalla de bienvenida ofrece una experiencia visual acorde con la identidad de la plataforma y facilita la navegación hacia las funcionalidades clave.



Ilustración 3. 46 Vista principal del usuario

El sistema permite a los alumnos registrarse en la plataforma y acceder mediante un sistema de autenticación, en el cual se verifican sus credenciales. Como se muestra en la captura de pantalla a continuación, los alumnos pueden iniciar sesión de manera eficiente y segura, y el sistema los redirige a sus páginas principales de su dashboard como usuario.



The registration form is titled "Registrar estudiante" and is located in the center of the page. It features a header with a logo and navigation links: "Home", "Quienes somos", "Contacto", and "Cuestionario". There are also "SIGN UP" and "LOG IN" buttons in the top right corner. The form itself includes fields for "Nombres", "Primer Apellido", "Segundo Apellido", "Género", "Edad", "País", "Tipo de usuario", "Semestre", "Matrícula", "Correo Electrónico", "Contraseña", and "Confirmar contraseña". At the bottom of the form are two buttons: "Regresar" (green) and "Registrarse" (yellow).

Registrar estudiante

Nombres

Primer Apellido Segundo Apellido

Género Edad

Seleccione un género Seleccione una edad

País

Seleccione un país

Tipo de usuario

Alumno

Semestre Matrícula

Semestre actual

Correo Electrónico

Contraseña

Confirmar contraseña

Regresar

Registrarse

Ilustración 3. 47 Registro para los alumnos



The login form is titled "Login" and is located in the center of the page. It features a header with a logo and navigation links: "Home", "Quienes somos", "Contacto", and "Cuestionario". There are also "SIGN UP" and "LOG IN" buttons in the top right corner. The form itself includes fields for "Correo electrónico o Matrícula" and "Contraseña". At the bottom of the form are two buttons: "Regresar" (green) and "Iniciar sesión" (yellow). Below the buttons is a link: "¿Olvidaste tu contraseña?".

Login

Correo electrónico o Matrícula:

Contraseña:

Regresar **Iniciar sesión**

[¿Olvidaste tu contraseña?](#)

Ilustración 3. 48 Inicio de sesión para los usuarios

El alumno, una vez ingresado al sistema, tiene acceso a los grupos en los que está inscrito. El sistema también permite ver los cuestionarios disponibles y los resultados obtenidos en cada uno. En la siguiente captura se observa cómo el alumno puede acceder a los cuestionarios resueltos y los puntajes obtenidos para cada uno.

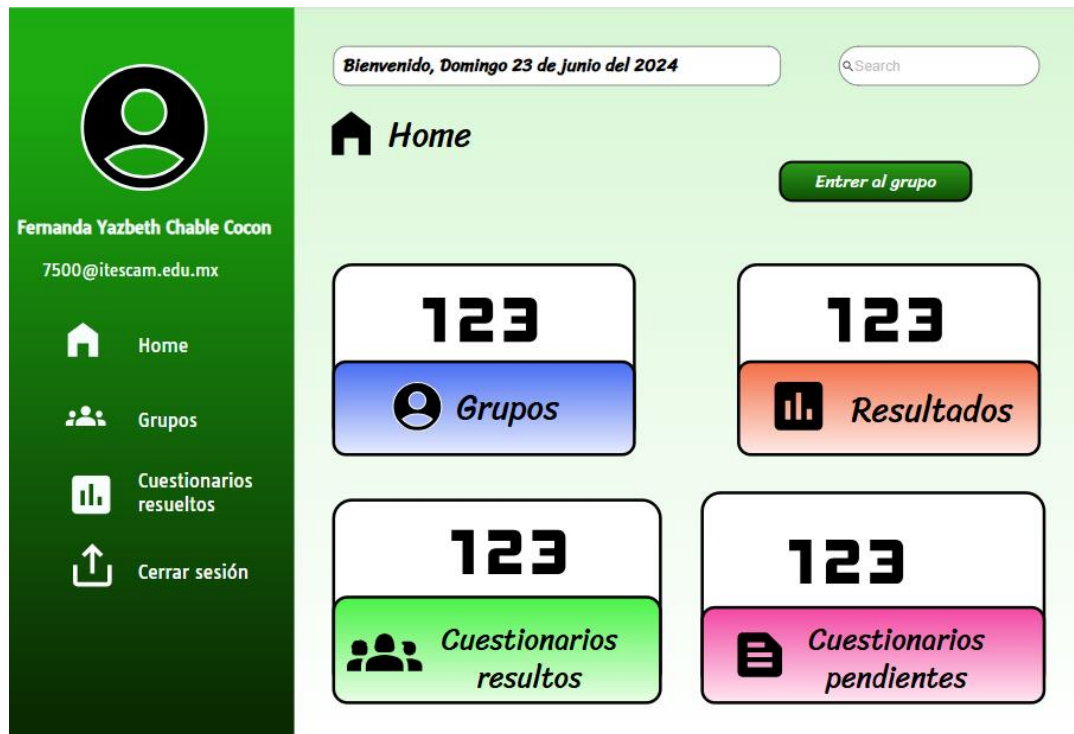


Ilustración 3. 49 Dashboard de alumnos



Ilustración 3. 50 Visualización de grupos del alumno

El alumno puede buscar un grupo con su código de acceso que le proporcione el maestro, con el botón de “Entrar al grupo”.

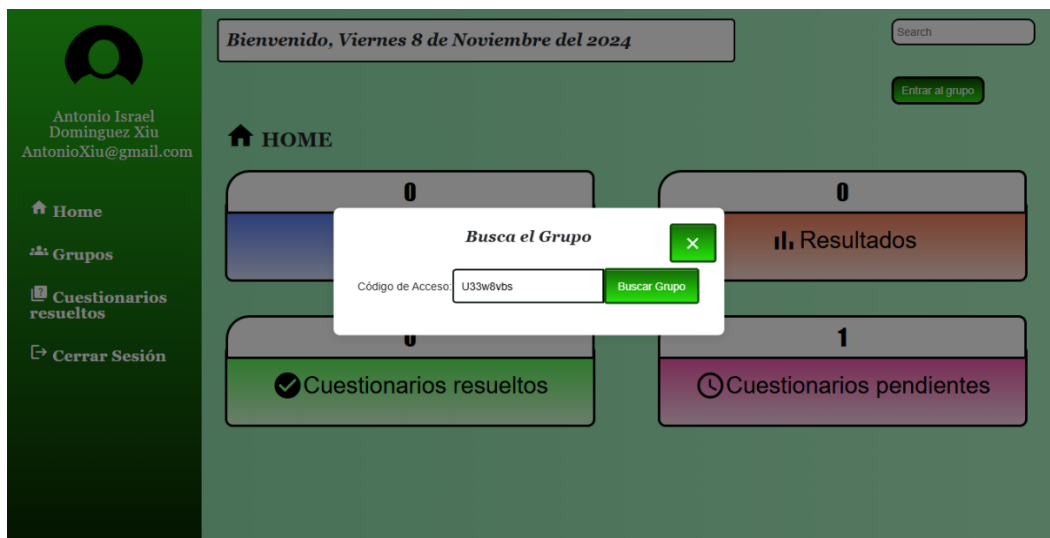


Ilustración 3. 51 Búsqueda de grupos

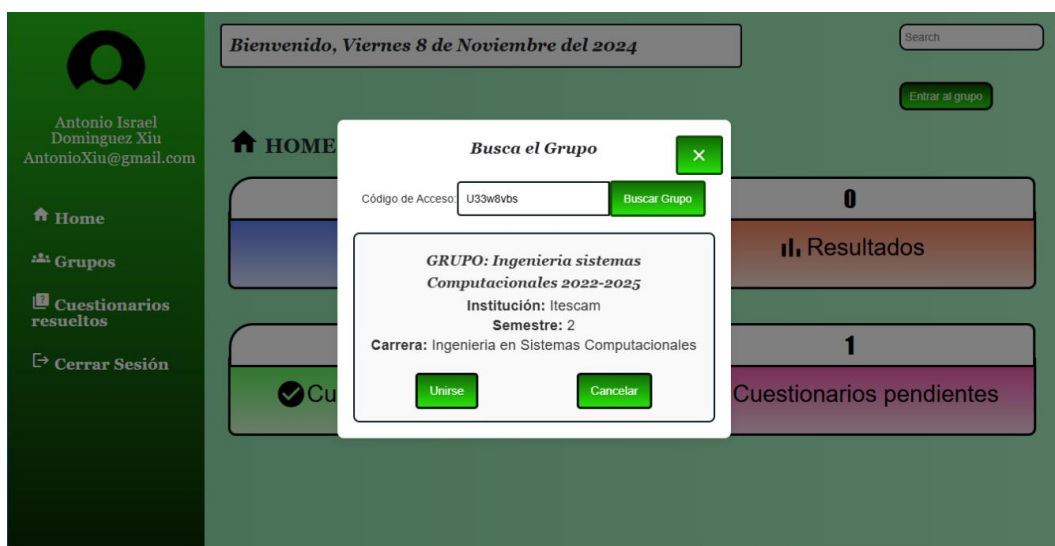


Ilustración 3. 52 Información del grupo encontrado



Ilustración 3. 53 Mensaje de éxito de ingreso al grupo

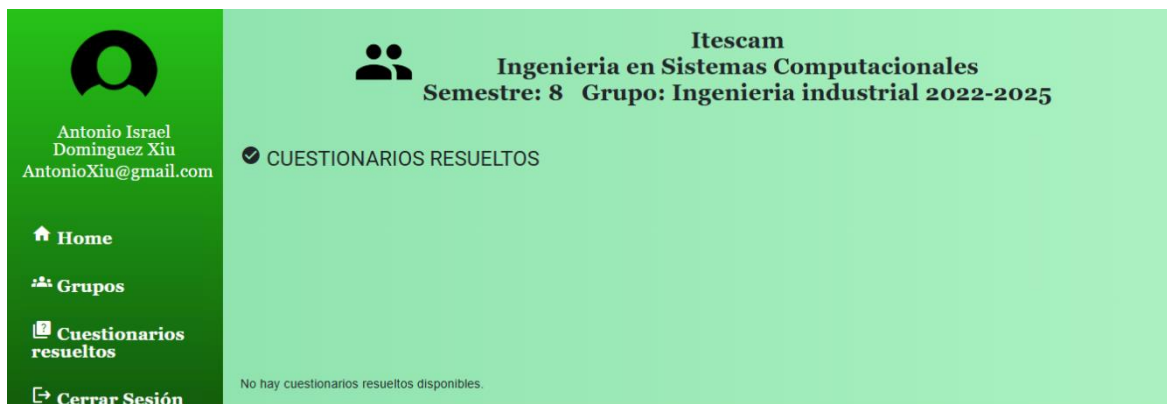


Ilustración 3. 54 Cuestionarios resueltos de un grupo (si no hay cuestionarios)

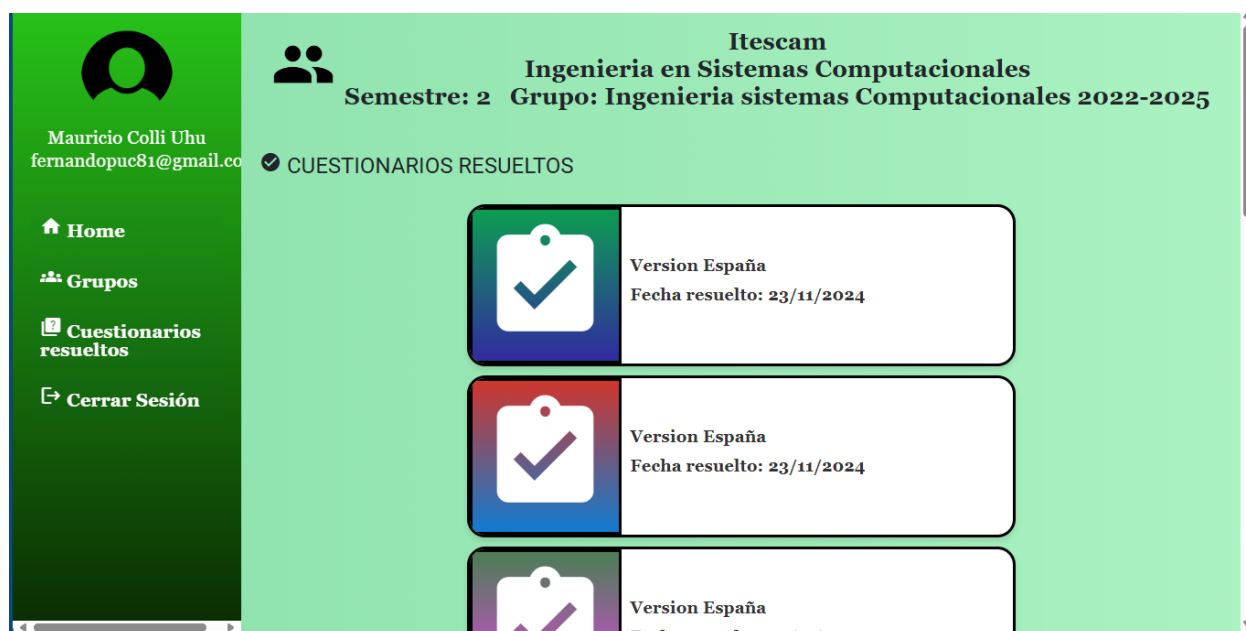


Ilustración 3. 55 Cuestionarios resueltos (Si ya respondió algún cuestionario)

El alumno puede visualizar sus cuestionarios pendientes

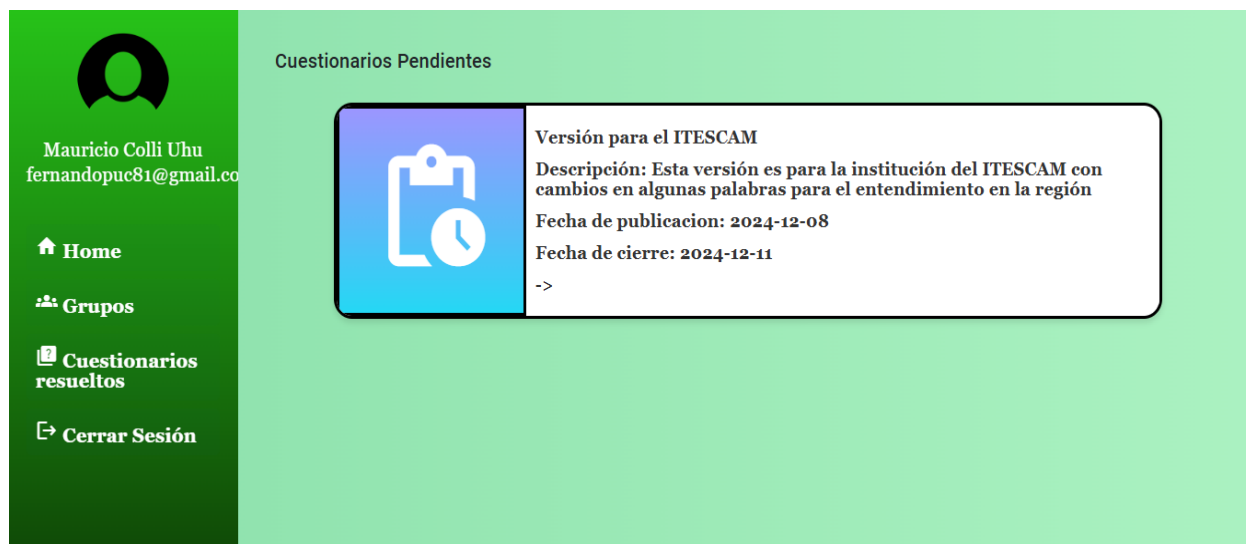


Ilustración 3. 56 Cuestionarios pendientes del alumno

En este módulo, el alumno tiene la posibilidad de ingresar al cuestionario, responder las preguntas y, al finalizar, obtener sus resultados de inmediato. Una vez completado el cuestionario, el sistema presenta una pantalla que muestra los siguientes detalles: la información personal del alumno, los datos relevantes del cuestionario como su nombre, fecha de realización y descripción de este.

Versión para el ITESCAM



Esta versión es para la institución del ITESCAM con cambios en algunas palabras para el entendimiento en la región

Me esmero en buscar cosas que necesitan hacerse.

☐ Nunca
☐ Raras veces
☐ Algunas veces
☐ Usualmente
☐ Siempre

Cuando me enfrento a un problema difícil invierto gran cantidad de tiempo en encontrar una solución.

☐ Nunca
☐ Raras veces
☐ Algunas veces
☐ Usualmente
☐ Siempre

Termino mi trabajo a tiempo.

☐ Nunca
☐ Raras veces

Ilustración 3. 57 Visualización del cuestionario

Termino mi trabajo a tiempo.

☐ Nunca
☐ Raras veces
☐ Algunas veces
☐ Usualmente
☐ Siempre

Me molesta cuando las cosas no se hace debidamente.

☐ Nunca
☐ Raras veces
☐ Algunas veces
☐ Usualmente
☐ Siempre

Prefiero situaciones en las que puedo controlar la mayor cantidad de la información al máximo.

☐ Nunca
☐ Raras veces
☐ Algunas veces
☐ Usualmente
☐ Siempre

Anterior

Siguiente

Ilustración 3. 58 Visualización del cuestionario



Yuridia Caamal Euan
caamal21@gmail.com

- Home
- Grupos
- Cuestionarios aplicados
- Alumnos
- Cerrar Sesión

Resultados de Angel Ek Xool (3680)

Nombre: Angel Ek Xool
Correo: AngelXool21@gmail.com
Semestre: 8°
Carrera: INGENIERIA EN SISTEMAS COMPUTACIONALES
Nombre del cuestionario: Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs)2
Descripción: Este cuestionario revisa en forma resumida las 10 características emprendedoras personales (CEPs), que forman la esencia de la personalidad emprendedora.
Fecha de publicación: 2024-11-20

Resultado de autoevaluación

Variable	Puntaje
Factor de corrección	17
Buscar y aprovechar las oportunidades	20
Ser Persistentes	23
Cumplimiento de compromisos	20
Exigir eficiencia y calidad	22
Correr riesgos moderados	20
Fijar Objetivos	25

Ilustración 3. 59 Información del alumno

A continuación, se despliega una tabla con los puntajes obtenidos en el cuestionario, desglosada por variables de las características emprendedoras. Además, se incluye una gráfica araña que visualiza de manera clara los resultados del alumno en comparación con los diferentes parámetros evaluados, lo que permite una interpretación rápida y visual de su desempeño.

Este módulo nos asegura que los resultados sean accesibles de inmediato, facilitando al alumno una retroalimentación completa y detallada de su rendimiento.



Yuridia Caamal Euan
caamal21@gmail.com

- Home
- Grupos
- Cuestionarios aplicados
- Alumnos
- Cerrar Sesión

Fecha de publicación: 2024-11-20

Resultado de autoevaluación

Variable	Puntaje
Factor de corrección	17
Buscar y aprovechar las oportunidades	20
Ser Persistentes	23
Cumplimiento de compromisos	20
Exigir eficiencia y calidad	22
Correr riesgos moderados	20
Fijar Objetivos	25
Buscar y conseguir información	19
Planificar y hacer seguimiento sistemático	19
Persuasión y creación de redes de apoyo	18
Tener autoconfianza	22

Gráfico de autoevaluación

Ilustración 3. 60 Tabla de puntajes



Yuridia Caamal Euan
caamal21@gmail.com

- Home
- Grupos
- Cuestionarios aplicados
- Alumnos
- Cerrar Sesión

Tener autoconfianza	22
---------------------	----

Gráfico de autoevaluación

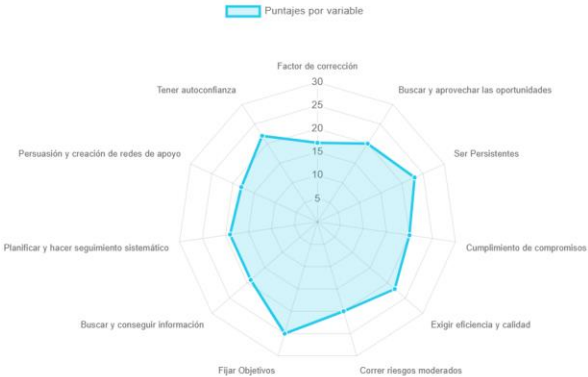


Ilustración 3. 61 Grafico de puntajes

En el módulo del Dashboard de Maestros, los usuarios con privilegios de docente tienen acceso a diversas funcionalidades clave para la gestión y monitoreo de los grupos de estudiantes y el progreso de los cuestionarios.

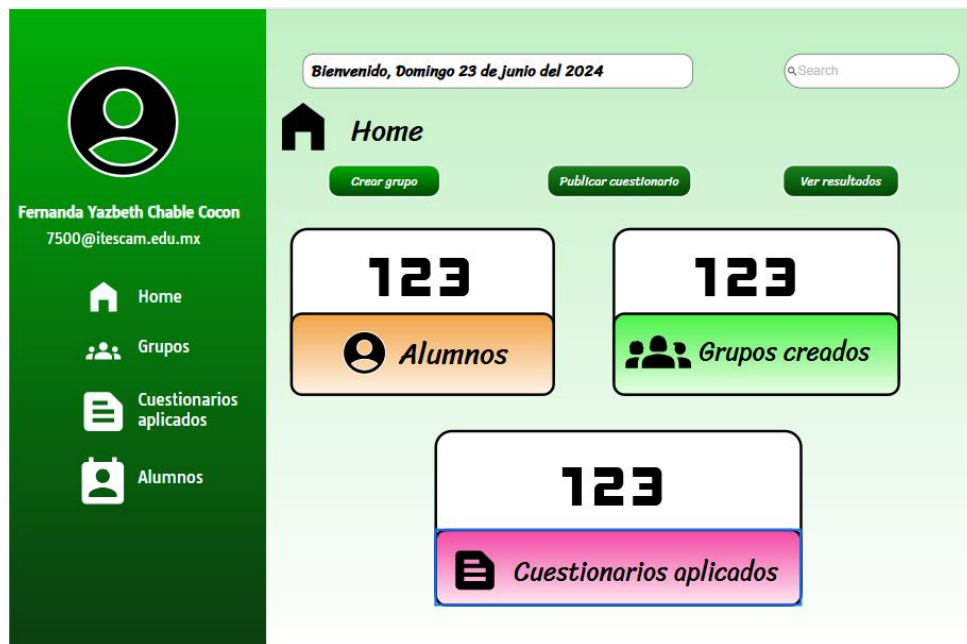


Ilustración 3. 62 Dashboard para maestros

Los maestros pueden ver los grupos de estudiantes que han creado, con detalles como el nombre del grupo, el semestre, la carrera y los alumnos inscritos.



Ilustración 3. 63 Visualización de grupos del maestro

Ellos tienen acceso a crear un grupo desde el botón “Crear grupo”, al ingresar tiene que llenar un formulario con los datos del grupo que desea crear.

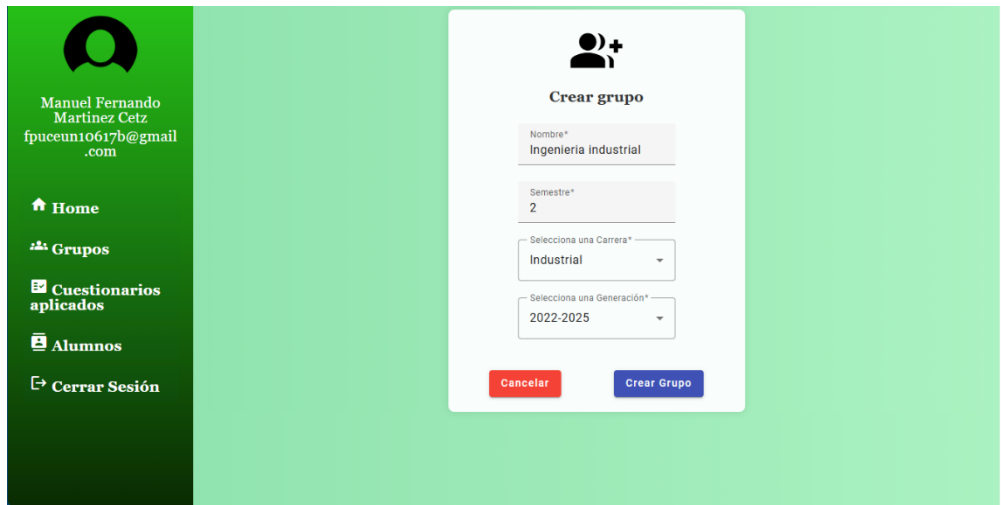


Ilustración 3. 64 Formulario para crear el grupo

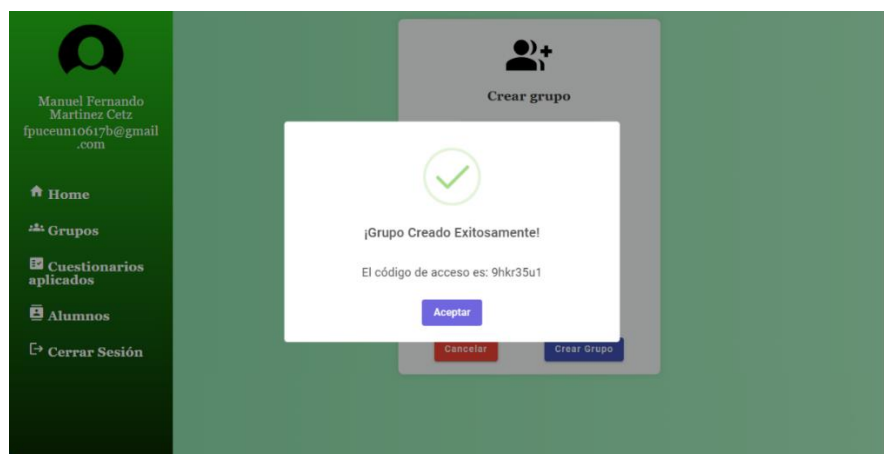


Ilustración 3. 65 Mensaje de éxito de grupo creado

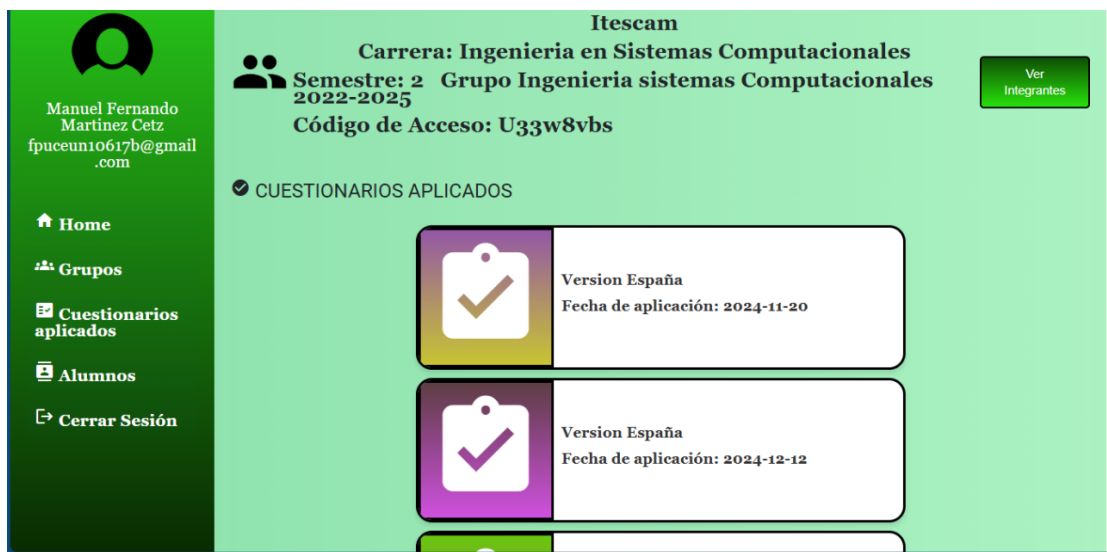


Ilustración 3. 66 Información del grupo

El maestro puede ver a los integrantes de los grupos con solo darle click al botón de “ver integrantes”

The screenshot shows a web application interface for 'Administración de Alumnos'. On the left is a green sidebar with a user profile for 'Manuel Fernando Martínez Cetz' and a menu with options: Home, Grupos, Cuestionarios aplicados, Alumnos, and Cerrar Sesión. The main content area has a title 'Administración de Alumnos' and a green button 'BUSCAR Y AÑADIR ALUMNO'. Below this is a table with student information.

NOMBRE	PRIMER APELLIDO	SEGUNDO APELLIDO	CARRERA	INSTITUCION	CORREO	SEMESTRE	ACCIONES
Manuel Fernando	Martinez	Cetz	Industrial	Itescam	fpuceun10617b@gmail.com	8°	
Mauricio	Colli	Uhu	Industrial	Itescam	fernandopuc81@gmail.com	6	

Ilustración 3. 67 Información de los integrantes del grupo

De igual manera pueden buscar y agregar a los alumnos, directamente desde el grupo que ha ingresado, con dar clic al botón de “Buscar y añadir alumno”

The screenshot shows the 'Busca al alumno' interface. It features a search bar with the text 'Denis' and a green 'Buscar alumno' button. Below the search bar, a modal window displays the search results for 'Itescam', showing the name 'Denis Alejandro Martinez Cetz' and the career 'Ingeniería en Sistemas Computacionales'. At the bottom of the modal are two buttons: 'Añadir alumno' (green) and 'Cancelar' (red).

Ilustración 3. 68 Búsqueda del alumno

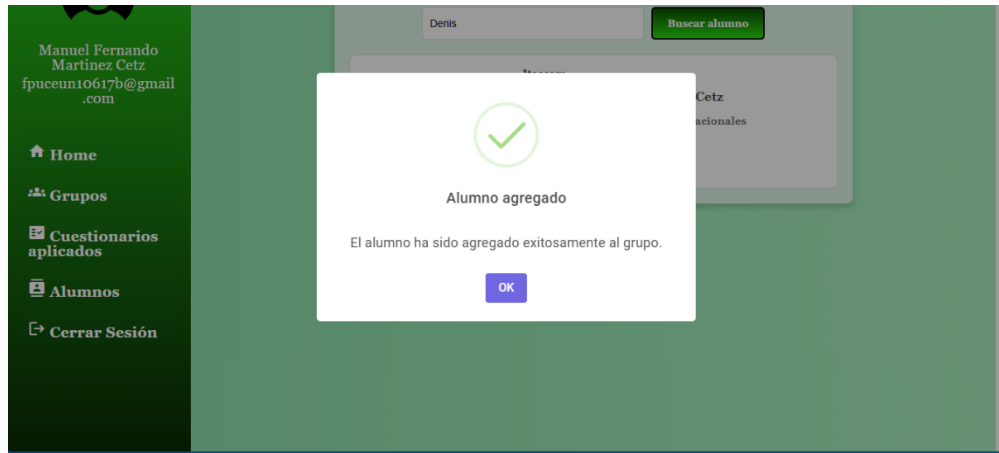


Ilustración 3. 69 Mensaje de éxito de alumno agregado al grupo

Al ingresar a la sección de cuestionarios aplicados se podrán observar todos los cuestionarios que han sido publicados por el docente. Con la información del cuestionario donde muestra el nombre del cuestionario, carrera, grupo, fecha de publicación y fecha de cierre.

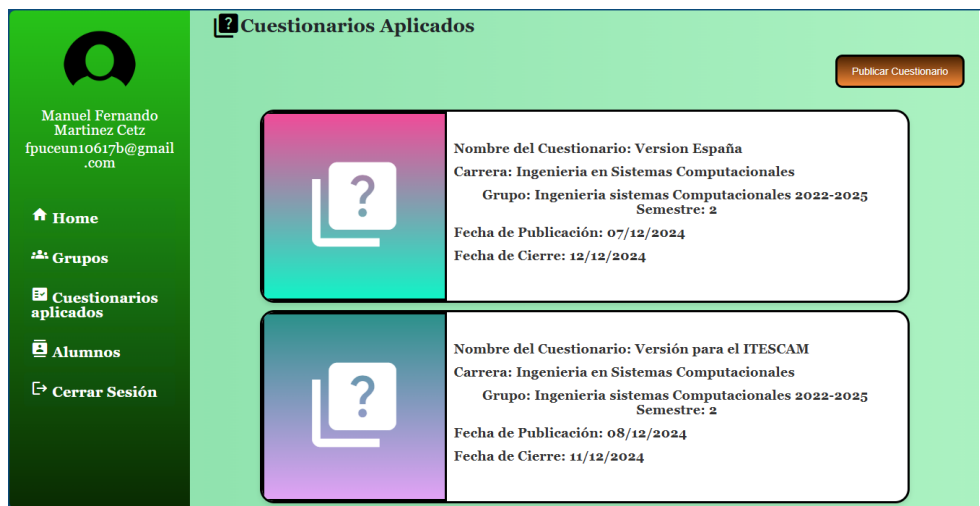


Ilustración 3. 70 Lista de cuestionarios aplicados

De igual manera tiene la opción de publicar un cuestionario ya existente ya sea en el mismo grupo o en uno diferente, al darle clic al botón de “publicar cuestionario” se mostrará una vista el cual el maestro podrá seleccionar la versión del cuestionario, el grupo al cual se aplicará, la fecha de publicación y la fecha de cierre.

Ilustración 3. 71 Visualización de publicación de cuestionario existente a un grupo

Al seleccionar el cuestionario se mostrarán todas las preguntas y opciones del cuestionario. Cabe mencionar que el cuestionario tiene un total de 55 preguntas por ende se podrán visualizar 10 preguntas por página.

Ilustración 3. 72 Formulario para publicar un cuestionario existente

Manuel Fernando
Martinez Cetz
fpuceun10617b@gmail
.com

Home
Grupos
Cuestionarios aplicados
Alumnos
Cerrar Sesión

☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
 3. Termino mi trabajo a tiempo.
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
 4. Me molesta cuando las cosas no se hace debidamente.
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
 5. Prefiero situaciones en las que puedo controlar la mayor cantidad de la información al máximo.
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
 6. Me gusta pensar sobre el futuro.
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
 7. Cuando comienza una tarea o un proyecto nuevo, reúno toda la información posible antes de darle curso.
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
 8. Planifico un proyecto grande dividiéndolo en tareas de menor tamaño.
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
 9. Logro que otros apoyen mis recomendaciones.
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
 10. Me siento confiado en que puede tener éxito en cualquier actividad que me propongo ejecutar.
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre

Anterior Siguiente
Publicar Cuestionario Cancelar

Ilustración 3. 73 Visualización del cuestionario a publicar (Cuestionario seleccionado)

Luego de publicar el cuestionario podrá aparecerá un mensaje el cual indicará que el cuestionario ha sido publicado correctamente.

Fecha de 09/12/20
Fecha de 23/12/2024

localhost:4200 dice
Cuestionario publicado exitosamente
Aceptar

Preguntas del Cuestionario

51. Me valgo de varias fuentes de información al buscar ayuda para llevar a cabo tareas o proyectos.
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
 52. Si no resulta un determinada estrategia para hacer frente a un problema, busco otro.
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
 53. Puedo lograr que personas con firmes convicciones y opiniones cambien de modo de pensar.
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
 54. Me mantengo firme en mis decisiones, aún cuando otras personas me contradigan enérgicamente.
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre
 55. Cuando no se algo no reparo en admitirlo.
☐ Nunca ☐ Raras veces ☐ Algunas veces ☐ Usualmente ☐ Siempre

Anterior Siguiente
Publicar Cuestionario Cancelar

Ilustración 3. 74 Mensaje cuestionario publicado

Al ingresar a la lista de cuestionarios aplicados puede visualizar a los integrantes del grupo y los cuestionarios aplicados en ese grupo en particular.



Manuel Fernando
Martínez Cetz
fpuceun10617b@gmail
.com

- Home
- Grupos
- Cuestionarios aplicados
- Alumnos
- Cerrar Sesión

Carrera: Ingeniería en Sistemas Computacionales
Grupo: Ingeniería sistemas Computacionales 2022-2025 **Semestre:** 2
Código de acceso: U33w8vbs

Integrantes del Grupo

Nombre	Primer Apellido	Segundo Apellido	Carrera
Manuel Fernando	Martínez	Cetz	Industrial
Mauricio	Colli	Uhu	Industrial
Denís Alejandro	Martínez	Cetz	Ingeniería en Sistemas Computacionales

Cuestionarios Aplicados

Nombre: Version España
Detalles de Cuestionario: Version de idioma español

Nombre: Versión para el ITESCAM
Detalles de Cuestionario: Esta versión es para la institución del ITESCAM con cambios en algunas palabras para el entendimiento en la región

[Ocultar Detalles](#)

Ilustración 3. 75 Información del cuestionario aplicado en el grupo

En este apartado el maestro podrá visualizar el nombre de la institución a la que pertenece el maestro, de igual manera su nombre y la lista de alumnos con los siguientes datos: Nombre, Apellidos, Edad, Género, Carrera y Grupo. Con 2 acciones, visualizar los resultados del alumno y eliminarlo del grupo.

ADMINISTRACIÓN DE ALUMNOS

Institución: Itescam

Profesor: Manuel Fernando Martínez Cetz

Lista de alumnos cargada

NOMBRE	PRIMER APELLIDO	SEGUNDO APELLIDO	MATRÍCULA	SEMESTRE	CORREO	EDAD	GÉNERO	CARRERA	GRUPO	ACCIONES
Mauricio	Colli	Uhu	7567	8	fernandopuc81@gmail.com	23	Masculino	Ingeniería en Sistemas Computacionales	Ingeniería sistemas Computacionales 2022-2025	 
Denís Alejandro	Martínez	Cetz	1234567890	8*	fsbdnj@gmail.com	30	Masculino	Ingeniería en Sistemas Computacionales	Ingeniería sistemas Computacionales 2022-2025	 

Ilustración 3. 76 Administración de alumnos del maestro

Cuestionarios Resueltos

Version España
[Ver Resultados](#)

Version España
[Ver Resultados](#)

Version España
[Ver Resultados](#)

Version España
[Ver Resultados](#)

Ilustración 3. 77 Visualización de cuestionarios resueltos del alumno

Al ingresar a “ver resultados” puede ver informes con las puntuaciones de los alumnos, detalladas por categoría y variable, así como las gráficas generadas, como la gráfica araña, para visualizar de manera más clara el rendimiento de cada estudiante.



Yuridia Caamal Euan
caamal21@gmail.com

- Home
- Grupos
- Cuestionarios aplicados
- Alumnos
- Cerrar Sesión

Resultados de Angel Ek Xool (3680)

Nombre: Angel Ek Xool
Correo: AngelXool21@gmail.com
Semestre: 8°
Carrera: INGENIERIA EN SISTEMAS COMPUTACIONALES
Nombre del cuestionario: Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs)2
Descripción: Este cuestionario revisa en forma resumida las 10 características emprendedoras personales (CEPs), que forman la esencia de la personalidad emprendedora.
Fecha de publicación: 2024-11-20

Resultado de autoevaluación

Variable	Puntaje
Factor de corrección	17
Buscar y aprovechar las oportunidades	20
Ser Persistentes	23
Cumplimiento de compromisos	20
Exigir eficiencia y calidad	22
Correr riesgos moderados	20
Fijar Objetivos	25

Ilustración 3. 78 Información del alumno



Yuridia Caamal Euan
caamal21@gmail.com

- Home
- Grupos
- Cuestionarios aplicados
- Alumnos
- Cerrar Sesión

Fecha de publicación: 2024-11-20

Resultado de autoevaluación

Variable	Puntaje
Factor de corrección	17
Buscar y aprovechar las oportunidades	20
Ser Persistentes	23
Cumplimiento de compromisos	20
Exigir eficiencia y calidad	22
Correr riesgos moderados	20
Fijar Objetivos	25
Buscar y conseguir información	19
Planificar y hacer seguimiento sistemático	19
Persuasión y creación de redes de apoyo	18
Tener autoconfianza	22

Gráfico de autoevaluación

Ilustración 3. 79 Tabla de puntajes por característica

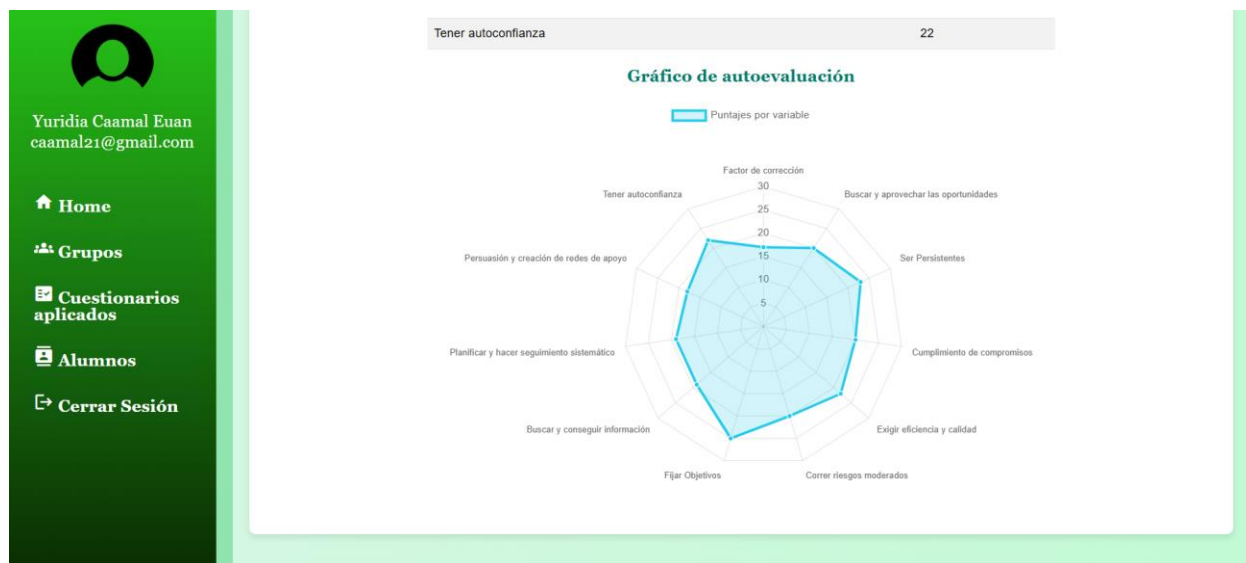


Ilustración 3. 80 Grafico de resultados

El Módulo de Administradores ha cumplido con todas las funcionalidades necesarias para la gestión completa de la plataforma, brindando a los administradores un control detallado sobre los aspectos más importantes del sistema. Las principales funcionalidades y los resultados obtenidos son los siguientes:

El administrador tiene la capacidad de gestionar a todos los usuarios del sistema, lo que incluye la creación de cuentas para maestros, estudiantes y otros roles. Esta funcionalidad permitió al administrador asignar roles y permisos de manera eficiente, asegurando que cada usuario tenga acceso a las secciones pertinentes.

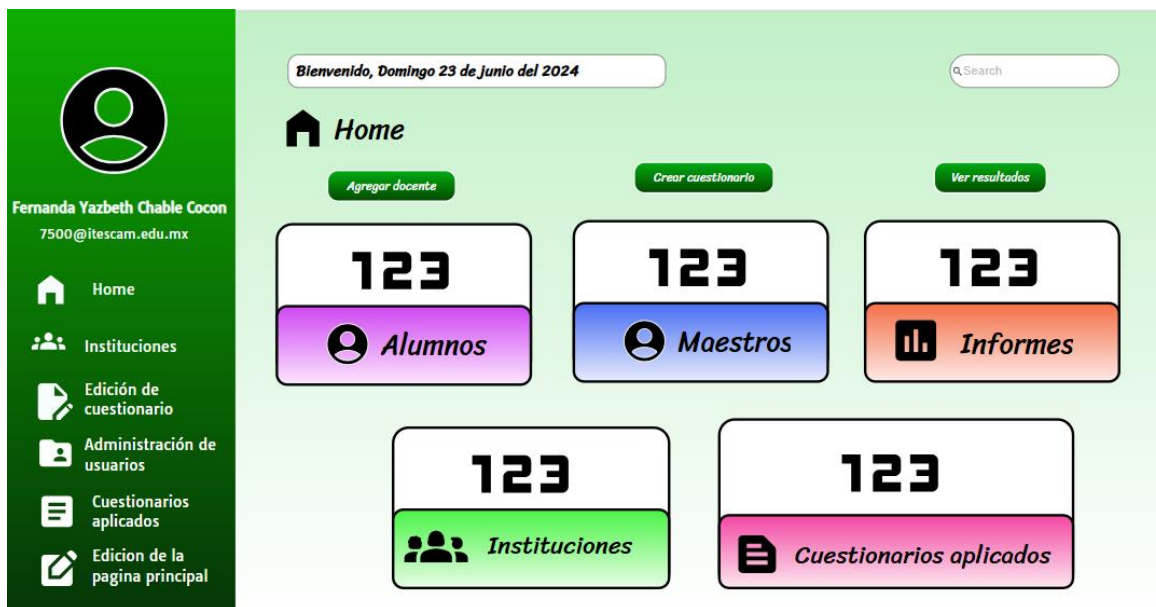


Ilustración 3. 81 Dashboard del administrador

Pueden asignar maestros a las instituciones correspondientes, lo que asegura una organización adecuada dentro de la plataforma. Esta característica ha permitido que el sistema se mantenga estructurado y alineado con las necesidades educativas de cada institución.



Ilustración 3. 82 Apartado de países

Al darle clic en el botón “Agregar País”, se desplegará una ventana para crear un nuevo país al sistema, al darle en agregar guardará el país al sistema.

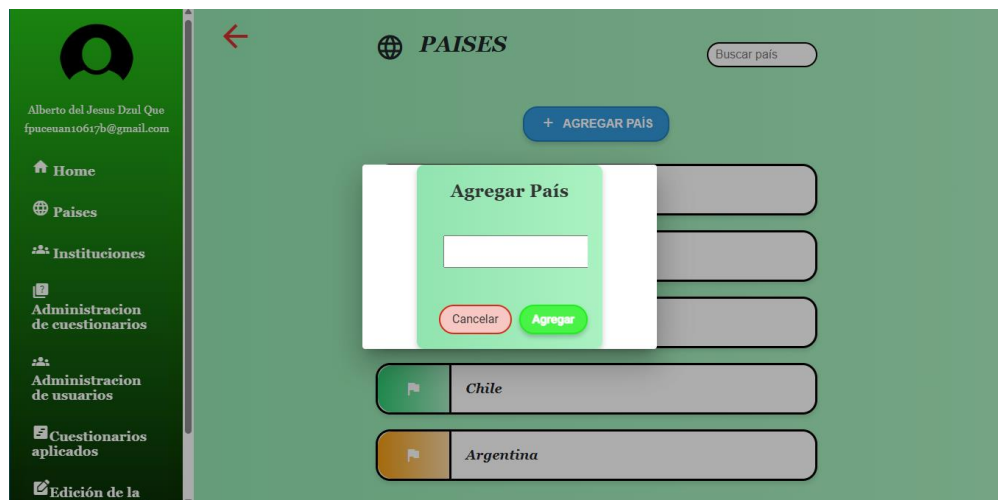


Ilustración 3. 83 Agregar país

De igual manera al darle clic a un país en específico mostrará las instituciones que pertenecen a ese país.



Ilustración 3. 84 Instituciones por país

Al ingresar a la institución podemos visualizar a los maestros y alumnos que hay en esa institución.



Ilustración 3. 85 Visualización de maestros y alumnos por institución

Al ingresar a maestros se visualizará la lista de maestros que pertenecen a la institución con la opción de agregar a un nuevo maestro, así como visualizar los grupos que pertenecen al maestro o eliminarlas de la institución.



Alberto del Jesus Dzul Que
fpuceuan10617b@gmail.com

- Home
- Instituciones
- Edición del cuestionario
- Administración de usuarios
- Cuestionarios aplicados
- Edición de la página principal
- Cerrar Sesión

←

 **Itescam**

 **Docentes**

Agregar docente

NOMBRE	PRIMER APELLIDO	SEGUNDO APELLIDO	CORREO	ACCIONES
Manuel Fernando	Martinez	Cetz	fpuceun10617b@gmail.com	 
Fernando Yahir	Puc	Euan	fpuceuan20618a@gmail.com	 
Fernanda	Yazbeth	Cocon	fernanda@gmail.com	 

Ilustración 3. 86 Lista de docentes por institución

Al darle clic al botón “agregar docente” redirigirá al administrador al formulario para añadir un nuevo docente con el país e institución preestablecido.



Registrar docente

Nombres

Primer Apellido

Segundo Apellido

Género

Seleccione un género ▾

Edad

Seleccione una edad ▾

País

Mexico ▾

Institución

Itescam ▾

Tipo de usuario

Maestro ▾

Folio

Correo Electrónico

Contraseña

Confirmar contraseña

Regresar

Registrarse

Ilustración 3. 87 Formulario de registro del docente

Al ingresar al icono de maestro en la lista nos dirigirá a los grupos que tiene los maestros



Ilustración 3. 88 Lista de grupos del docente

Luego de ingresar al grupo se podrán observar los integrantes, junto con la información del grupo.



Ilustración 3. 89 Lista de integrantes del grupo

Ahora regresando al apartado de administración de usuarios por institución, el maestro podrá visualizar la lista de todos los alumnos o filtrada por carrera.



The screenshot shows a web application interface for 'Itescam'. On the left is a green sidebar with a user profile at the top (Alberto del Jesus Dzul Que, fpucuan10617b@gmail.com) and a menu with options: Home, Instituciones, Edición del cuestionario, Administración de usuarios (highlighted), Cuestionarios aplicados, Edición de la página principal, and Cerrar Sesión. The main content area has a light green background. At the top, it says 'Itescam' and 'Alumnos'. Below this is a dropdown menu currently set to 'Ingeniería en Sistemas'. A table displays a list of students with columns for NOMBRE, PRIMER APELLIDO, SEGUNDO APELLIDO, CARRERA, INSTITUCIÓN, and CORREO. Two students are listed: Denis Alejandro Martinez Celz and Antonio Israel Dominguez Xiu, both from 'Itescam' and studying 'Ingeniería en Sistemas Computacionales'. A scroll bar is visible at the bottom of the table.

NOMBRE	PRIMER APELLIDO	SEGUNDO APELLIDO	CARRERA	INSTITUCIÓN	CORREO
Denis Alejandro	Martinez	Celz	Ingeniería en Sistemas Computacionales	Itescam	tsbdri@gmail.com
Antonio Israel	Dominguez	Xiu	Ingeniería en Sistemas Computacionales	Itescam	AntonioXiu@gmail.com

Ilustración 3. 90 Lista de alumnos de la institución

En este apartado al dar clic en el botón “ver detalles” el administrador podrá ver la información del cuestionario, se visualiza el nombre, la descripción, las fechas de publicación por grupo. De igual manera todas las preguntas con su signo y variables los cuales se visualizarán 10 preguntas por página incluyendo un botón para regresar a la lista de cuestionarios



The screenshot shows a web application interface for 'Version España'. The left sidebar is identical to the previous screenshot, with 'Administración de cuestionarios' highlighted. The main content area has a light gray background. At the top, it says 'Version España'. Below this is a 'Descripción: Version de idioma español'. A section titled 'Fechas por Grupo:' contains a text box with 'Grupo: Ingeniería sistemas Computacionales 2022-2025 Publicación: 12/7/24'. A section titled 'Preguntas' lists three questions. Each question has a 'Signo: Positiva' and a 'Variables:' section. The first question is '1. Me esmero en buscar cosas que necesitan hacerse.' with a variable 'Iniciativa'. The second question is '2. Cuando me enfrente a un problema difícil invierto gran cantidad de tiempo en encontrar una solución.' with a variable 'Persistente'. The third question is '3. Terminó mi trabajo a tiempo.' with a variable 'Persistente'. A scroll bar is visible on the right side of the page.

Ilustración 3. 91 Detalles del cuestionario

De igual manera el administrador puede crear una nueva versión del cuestionario, asignando el nombre de la versión y su descripción. por ende, al ingresar a esta sección se muestran las preguntas del cuestionario original ya que se crea una nueva versión de este. De igual manera se solicitó que solamente se pueda modificar las preguntas y el signo de la pregunta y las variables se mantienen intactas para respetar el formato del cuestionario.

CREAR UN NUEVO CUESTIONARIO

Nombre
Versión Tecnológico de Champotón

Descripción
Esta versión se realiza exclusivamente para el ITESCHAM

PREGUNTAS DEL CUESTIONARIO

Pregunta	Variables	Signo
Pregunta 1 Me esmero en buscar cosas que neces	Iniciativa	Positiva
Pregunta 2 Cuando me enfrento a un problema difi	Persistente	Positiva
Pregunta 3 Termino mi trabajo a tiempo.	Cumplimiento	Positiva
Pregunta 4 Me molesta cuando las cosas no se ha	Eficiencia	Negativa

Ilustración 3. 92 Crear nueva versión del cuestionario

En este apartado se muestran las instrucciones del cuestionario que observará el alumno.

Autoevaluación de las Características Emprendedoras Personales (CEPs)

Instrucciones

Bienvenido a la Autoevaluación de las Características Emprendedoras Personales (CEPs). Antes de comenzar, por favor, lee atentamente las siguientes instrucciones:

- El cuestionario consta de 55 declaraciones breves. Lea cuidadosamente cada una y decida qué tan acertadamente le describe. Sea honesto consigo mismo. Recuerde que nadie tiene puntaje perfecto. Este es un autoanálisis, por lo tanto el único que juzgará su resultado es usted mismo.
- Seleccione el número que corresponde para designar la media en que cada declaración describe su comportamiento:

1= Nunca	2= Raras veces	3= Algunas veces	4= Usualmente	5= Siempre
----------	----------------	------------------	---------------	------------

- Algunas declaraciones pueden ser similares, pero ninguna es igual, por lo tanto deben ser respondidas todas.
- Si tiene duda sobre alguna, pase a la siguiente, y al finalizar vuelva sobre las que dejó sin contestar.

[Cancelar](#) [Empezar Cuestionario](#)

Ilustración 3. 93 Vista de instrucciones del cuestionario

Una de las funcionalidades clave para los administradores es la capacidad de generar informes detallados sobre los resultados de los estudiantes. Los informes permiten el análisis de los puntajes y el rendimiento de los estudiantes, proporcionando datos valiosos para la toma de decisiones y el mejoramiento continuo del sistema educativo.

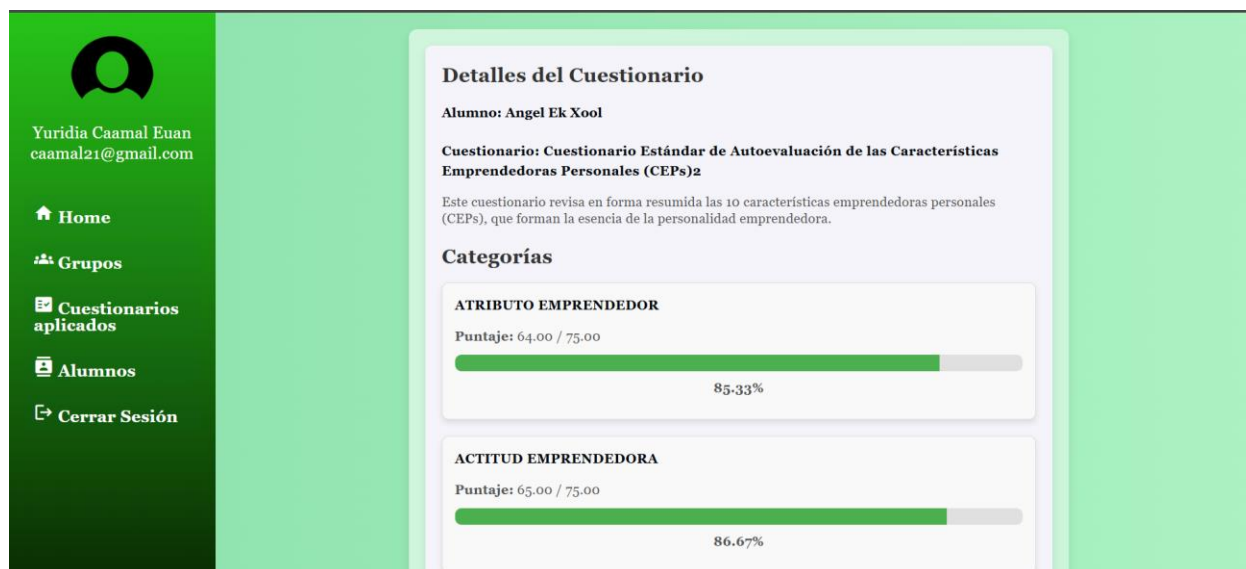


Ilustración 3. 94 Análisis macro



Ilustración 3. 95 Análisis macro (Puntajes por categorías)

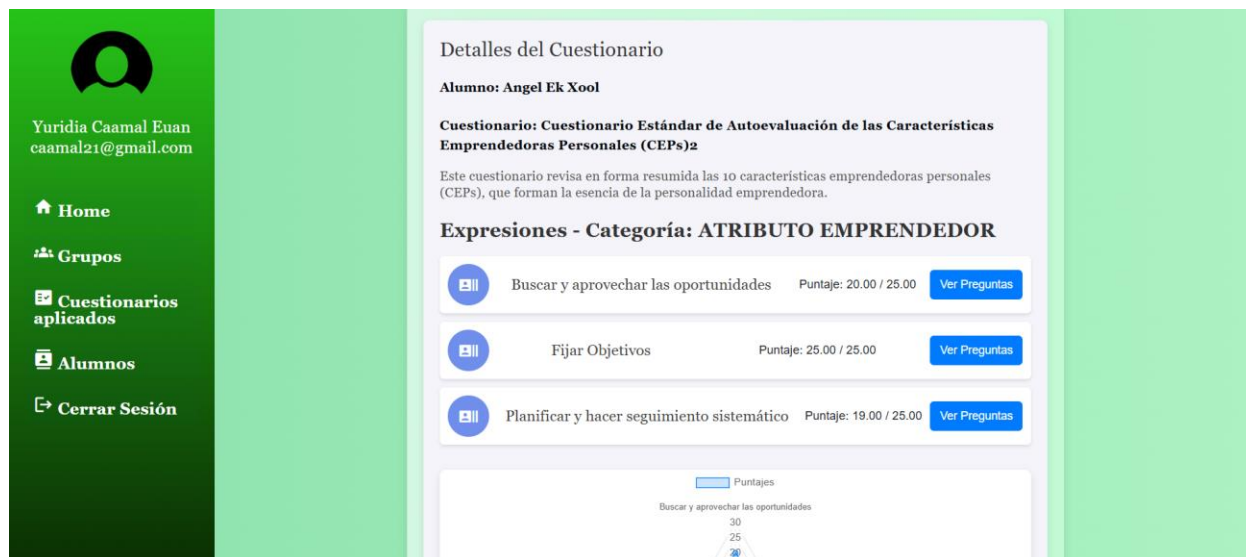


Ilustración 3. 96 Análisis micro (Puntajes por variables de cada categoría)



Ilustración 3. 97 Análisis micro (Puntajes por variables de cada categoría)



Yuridia Caamal Euan
caamal21@gmail.com

- Home
- Grupos
- Cuestionarios aplicados
- Alumnos
- Cerrar Sesión

Detalles del Cuestionario

Alumno: Angel Ek Xool

Cuestionario: Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs)2

Este cuestionario revisa en forma resumida las 10 características emprendedoras personales (CEPs), que forman la esencia de la personalidad emprendedora.

Preguntas - Expresión: Buscar y aprovechar las oportunidades

Pregunta	Respuesta del alumno	Puntaje máximo
Me esmero en buscar cosas que necesitan hacerse.	4	5
Hago lo que se necesita hacer sin que otros tengan que pedirme que lo haga.	5	5
Me gustan los desafíos y nuevas oportunidades.	5	5
Prefiero desempeñar tareas que domino a la perfección y en las que me siento seguro.	5	5
Me aventura a hacer cosas nuevas y diferentes de lo que he realizado en el pasado.	5	5

Ilustración 3. 98 Análisis micro (Puntajes por pregunta de cada variable



Yuridia Caamal Euan
caamal21@gmail.com

- Home
- Grupos
- Cuestionarios aplicados
- Alumnos
- Cerrar Sesión

Resultados de Angel Ek Xool (3680)

Nombre: Angel Ek Xool
Correo: AngelXool21@gmail.com
Semestre: 8°
Carrera: INGENIERIA EN SISTEMAS COMPUTACIONALES
Nombre del cuestionario: Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs)2
Descripción: Este cuestionario revisa en forma resumida las 10 características emprendedoras personales (CEPs), que forman la esencia de la personalidad emprendedora.
Fecha de publicación: 2024-11-20

Resultado de autoevaluación

Variable	Puntaje
Factor de corrección	17
Buscar y aprovechar las oportunidades	20
Ser Persistentes	23
Cumplimiento de compromisos	20
Exigir eficiencia y calidad	22
Correr riesgos moderados	20
Fijar Objetivos	25

Ilustración 3. 99 Datos del alumno (Puntajes en general)



Fecha de publicación: 2024-11-20

Resultado de autoevaluación

Variable	Puntaje
Factor de corrección	17
Buscar y aprovechar las oportunidades	20
Ser Persistentes	23
Cumplimiento de compromisos	20
Exigir eficiencia y calidad	22
Correr riesgos moderados	20
Fijar Objetivos	25
Buscar y conseguir información	19
Planificar y hacer seguimiento sistemático	19
Persuasión y creación de redes de apoyo	18
Tener autoconfianza	22

Gráfico de autoevaluación

Ilustración 3. 100 Análisis de puntajes en general



Tener autoconfianza 22

Gráfico de autoevaluación



Ilustración 3. 101 Grafico de puntajes

El módulo de Análisis e Informes está diseñado para procesar los datos obtenidos a través del cuestionario de Características Emprendedoras Personales (CEPs) y generar reportes que permitan identificar perfiles emprendedores en egresados universitarios. Aunque el diseño del módulo está bien establecido, los controladores y funciones específicas para realizar el análisis avanzado y la generación de informes aún están en desarrollo (Anexo 111-118). Este aspecto sigue siendo una prioridad en el proyecto.

Actualmente, el módulo tiene el objetivo de incluir análisis como la identificación de fortalezas y áreas de mejora en las características emprendedoras de los egresados, comparaciones grupales y generación de gráficas y tablas dinámicas. Estos informes permitirán a las instituciones educativas comprender mejor las capacidades emprendedoras de sus egresados y tomar decisiones fundamentadas para apoyar su desarrollo profesional.

El trabajo en esta área continúa, y se están implementando metodologías que aseguren la calidad y precisión de los análisis. Este módulo representa una de las áreas de mejora más significativas del sistema y se espera que, una vez completado, brinde una herramienta valiosa para la evaluación de perfiles emprendedores.

3.9. PRUEBAS EXPLORATORIAS

Las pruebas exploratorias se realizaron en el sistema con el objetivo de evaluar el comportamiento del sistema sin un guion predefinido, explorando libremente las funcionalidades clave y al mismo tiempo documentar cualquier hallazgo relevante.

3.9.1. Registro de Pruebas Exploratorias.

ID	Módulo	Objetivo de la Prueba	Acción realizada	Resultados observados	Notas/ Errores encontrados
PE01	Gestión de usuarios	Probar la creación de nuevos docentes	Cree un nuevo docente desde la plataforma de administrados	El usuario se creó correctamente	Sin errores
PE02	Asignación de cuestionarios	Intentar asignar un cuestionario sin seleccionar un grupo	Hice clic en "Grupo" en el formulario sin elegir un grupo	Mensaje de error indicando que falta seleccionar un grupo	Correcto
PE03	Exportación de informes	Probar exportación de informes a PDF	Intente exportar un informe estadístico de un grupo desde la sección de "estadísticas del grupo"	El archivo PDF se descargó correctamente y contenía la información completa y correcta	Correcto
PE04	Comparación de Estadísticas	Evaluar la comparación entre dos grupos	Seleccione dos grupos para comparar	La grafica de araña se actualizo correctamente con información del grupo	Sin errores
PE05	Gestión de perfil del administrador	Verificar que el administrador pueda gestionar carreras países e instituciones	Cree una nueva carrera y una institución desde el perfil de administrador	Los datos fueron guardados correctamente	Sin errores
PE06	Consulta de resultados de cuestionarios	Validar que el administrador pueda ver los resultados de	Ingresé a la sección de administración de usuarios,	Se mostraron correctamente	Sin errores

		<i>todos los cuestionarios de los alumnos</i>	<i>accedí a la lista de alumnos, para seleccionar la acción “resultados”, accedí a una lista de los resultados del alumno</i>		
PE07	<i>Creación y gestión de grupos</i>	<i>Validar la creación de grupos con los distintos parámetros a través de un formulario</i>	<i>Cree un grupo con un nombre, carrera y generación aginada.</i>	<i>El grupo se creó correctamente</i>	<i>Sin errores</i>
PE08	<i>Creación de cuestionarios</i>	<i>Verificar que el administrador pueda crear un cuestionario para que el maestro lo pueda publicar en un grupo</i>	<i>Cree un cuestionario desde el perfil de administrador</i>	<i>El cuestionario fue guardado correctamente y está disponible para ser publicado por el cuestionario</i>	<i>Sin errores</i>
PE09	<i>Navegación del menú principal</i>	<i>Verificar que el menú principal muestre seis botones y que cada uno dirige a la sección correspondiente</i>	<i>Ingresa a la página principal y revise los botones Home, Quienes somos, Contacto, Cuestionario, Registro e Inicio de sesión</i>	<i>Cada botón redirigió correctamente a la página correspondiente</i>	<i>Sin errores</i>
PE10	<i>Inicio de sesión</i>	<i>Validar que un usuario puede iniciar sesión con credenciales correctas</i>	<i>Ingresa correo y contraseña válidos y presione “Iniciar sesión”</i>	<i>El usuario fue redirigido a su perfil correctamente</i>	<i>Sin errores</i>
PE11	<i>Registro y acceso a la plataforma (Alumno)</i>	<i>Validar que un alumno puede registrarse y</i>	<i>Completé el formulario de registro con datos válidos y</i>	<i>El alumno pudo acceder a su pantalla de inicio correctamente</i>	<i>Sin errores</i>

		<i>acceder correctamente</i>	<i>luego inicié sesión</i>		
PE12	<i>Visualización de cuestionarios pendientes</i>	<i>Validar que el alumno pueda ver sus cuestionarios pendientes</i>	<i>Ingresa al apartado de “cuestionarios pendientes” y seleccione un cuestionario</i>	<i>Se mostraron correctamente los cuestionarios pendientes</i>	<i>Sin errores</i>
PE13	<i>Responder cuestionarios</i>	<i>Validar que el alumno pueda completar y enviar un cuestionario</i>	<i>Accedí a un cuestionario, contesté las preguntas y lo envié.</i>	<i>El cuestionario fue enviado correctamente</i>	<i>Sin errores</i>
PE14	<i>Retroalimentación personalizada</i>	<i>Validar que el alumno reciba retroalimentación tras completar un cuestionario</i>	<i>Completé un cuestionario y envié las respuestas</i>	<i>Se mostro correctamente la retroalimentación con puntajes y la grafica</i>	<i>Sin errores</i>
PE15	<i>Ver progreso individual</i>	<i>Verificar que el alumno pueda visualizar su progreso en la plataforma</i>	<i>Accedí al dashboard del estudiante y revisé su progreso en cuestionarios completados</i>	<i>El progreso fue visible y preciso</i>	<i>Sin errores</i>
PE16	<i>Informe Macro, Intermedio, Micro y General de Resultados</i>	<i>Validar que el maestro pueda ver los puntajes de los alumnos de manera detallada</i>	<i>Ingresé al dashboard del maestro y accedí a la administración de alumnos</i>	<i>Los puntajes se mostraron de manera clara y ordenada, los puntajes y la gráfica son correctas.</i>	<i>Sin errores</i>

Tabla 3. 5 Tabla de pruebas exploratorias

3.10. CASO DE PRUEBAS DE USABILIDAD.

En esta sección se detallan los escenarios de prueba para validar los requerimientos funcionales del sistema. Se incluyen pruebas para los diferentes módulos y funcionalidades del sistema, así como la visualización de estadísticas de grupo, la comparación de grupos, la gestión de usuarios y exportación de informes en PDF.

3.10.1. Módulo de Bienvenida.

Escenario
Navegación del Menú Principal
ID: TC01

Descripción:		Verificar que el menú principal muestra seis botones y que cada uno dirige a la sección correspondiente.			
Requerimientos Asociados:		R01			
Tarea		Resultado Esperado	Resultado	Estado (Aprobado/Fallido)	Observaciones
1	Ingresar a la página principal.	Cada botón redirige correctamente a la página correspondiente.	Se envió a la página principal y se ingresaron a todos los sitios al darle clic.	Aprobado.	Sin errores.
2	Verificar que los botones Home, Quiénes somos, Contacto, Cuestionario, Registro e Inicio de sesión están presentes.				
3	Hacer clic en cada botón y verificar que lleva a la sección correcta.				

Tabla 3. 6 Caso de Prueba de Usabilidad Menú Principal

3.10.2. Módulo de Inicio de Sesión.

Escenario					
Acceso con Credenciales Validas					
ID: TC02					
Descripción:		Validar que un usuario puede iniciar sesión con credenciales correctas.			
Requerimientos Asociados:		R08			
Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Ingresar correo y contraseña válidos.	El usuario es redirigido a su perfil.	Envía a la página principal después de ingresar los datos creados anteriormente y darle clic a "iniciar sesión".	Aprobado	Sin errores.
2	Hacer clic en "Iniciar sesión".				

Tabla 3. 7 Caso de Prueba de Usabilidad Inicio de Sesión

3.11.3. Módulo de Perfil de Alumno.

Escenario
Registro y Acceso a la Plataforma

ID: TC03					
Descripción:		El alumno debe ser capaz de registrarse y acceder a la plataforma			
Requerimientos Asociados:		R11			
Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Ir a la página de registro.	El alumno debe ser registrado y acceder correctamente a la plataforma, a su pantalla de inicio de su dashboard.	Se creó mi propio usuario ingresando los datos solicitados los cuales permiten ingresar a la página principal sin detalles.	Aprobado	Sin errores
2	Ingresar los datos requeridos (nombre, apellido, correo electrónico, etc.)				
3	Hacer clic en “Registrarse”				
4	Ingresar con el correo electrónico y la contraseña recién creada				

Tabla 3. 8 Caso de Prueba de Usabilidad Perfil del Alumno

Escenario					
Ver Cuestionarios Asignados o Pendientes					
ID: TC04					
Descripción:		El alumno puede ver los cuestionarios asignados por su maestro o que tenga pendiente por resolver			
Requerimientos Asociados:		R12			
Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Ingresar a la plataforma	El alumno debe poder encontrar con facilidad el apartado de los cuestionarios pendientes y poder ingresar a ellos correctamente	Se ingresó a la plataforma con el usuario creado, apareció los cuestionarios a resolver y se accedió a ellos.	Aprobado	Sin errores
2	Al estar en el perfil del estudiante				
3	Hacer clic en el apartado que diga “Cuestionarios Pendientes”				
4	Al ingresar al apartado, selecciona el cuestionario que quieras resolver.				

Tabla 3. 9 Caso de Prueba de Usabilidad Cuestionarios Pendientes

Escenario					
Responder Cuestionarios					
ID: TC05					

Descripción:		El alumno debe poder responder los cuestionarios, así como enviar sus respuestas.			
Requerimientos Asociados:		R13			
Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Acceder al cuestionario que quisiera resolver	El alumno debe poder completar y enviar el cuestionario correctamente.	El cuestionario se contestó correctamente y fue enviado sin detalles.	Aprobados	Sin errores.
2	Contestar las preguntas				
3	Enviar el cuestionario				

Tabla 3. 10 Caso de Prueba de Usabilidad Respuesta a cuestionarios

Escenario					
Retroalimentación personalizada					
ID: TC06					
Descripción:		El alumno debe poder recibir su retroalimentación a su propia autoevaluación después de haber completado el cuestionario.			
Requerimientos Asociados:		R14			
Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Completar el cuestionario	El alumno debe ver su puntaje en cada una de las características y cualidades emprendedoras que conforman el cuestionario.	Arrojó el puntaje obtenido, así como la gráfica de comparación.	Aprobado	Sin errores
2	Enviar el cuestionario				
3	Ver la retroalimentación de su autoevaluación.				

Tabla 3. 11 Caso de Prueba de Usabilidad Retroalimentación Personalizada

Escenario					
Ver Progreso Individual					
ID: TC07					
Descripción:		El alumno debe poder ver su progreso a lo largo de los cuestionarios.			
Requerimientos Asociados:		R15			
Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Ingresar al dashboard del estudiante	El progreso debe ser visible y preciso	Se accedió a los resultados de los cuestionarios	Aprobado	Sin errores
2	Ver los resultados y progreso de los				

	cuestionarios completados		resueltos y de los cuestionarios pendientes.		
--	---------------------------	--	--	--	--

Tabla 3. 12 Caso de Prueba de Usabilidad Progreso Individual

3.10.3. Módulo de Perfil de Maestro.

Escenario					
Asignación de Cuestionarios a Grupos					
ID: TC08					
Descripción:		Verificar que los maestros pueden asignar cuestionarios a grupos específicos.			
Requerimientos Asociados:		R17, R18, R35			
Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Iniciar sesión como maestro.	El cuestionario es asignado correctamente y aparece en la lista de cuestionarios del grupo.	Se inició sesión y fue fácil asignar el cuestionario al grupo en cuestión.	Aprobado.	Sin errores.
2	Acceder a la sección de gestión de cuestionarios.				
3	Seleccionar un grupo y asignar un cuestionario.				
4	Establecer una fecha de inicio y fin para un cuestionario				
5	Visualizar las preguntas y respuestas que conforman el cuestionario				
4	Publicar cuestionario al grupo				

Tabla 3. 13 Caso de Prueba de Usabilidad Perfil de Docentes

Escenario					
Informe Macro, Intermedio, Macro y General de los Resultados de los Alumnos					
ID: TC09					
Descripción:		El maestro puede ver los puntajes obtenidos en los cuestionarios de los alumnos de manera detallada			
Requerimientos Asociados:		R19, R20			
Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Ingresar al dashboard del maestro	Los puntajes de los alumnos se	Los resultados	Aprobado	Sin errores

2	Ir a la sección de administración de alumnos	deben de mostrar de forma clara y ordenada	de los alumnos fueron mostrados de manera clara y ordenada.		
3	Seleccionar los resultados de un alumno.				
4	Ingresar a la lista de resultados generados del alumno por los cuestionarios que ha contestado				
5	Seleccionar "Resultados Generales" o "Resultados Detallados" dependiendo de qué manera queramos visualizar los puntajes				
6	Ver los puntajes de los alumnos de manera general o bien de manera detallada (Macro = categorías, Intermedio = variables por categorías, Micro = Preguntas por variable)				

Tabla 3. 14 Caso de Prueba de Usabilidad Vistas de Resultados

Escenario					
Exportar Informes o Resultados de un Cuestionario					
ID: TC10					
Descripción:		El maestro puede exportar los informes o resultados de un alumno en formato PDF			
Requerimientos Asociados:		R21			
Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Ingresar al dashboard del maestro	El informe debe ser exportado correctamente en el formato seleccionado.	Las comparaciones fueron exportadas correctamente al formato	Aprobado	Sin errores.
2	Ir a la sección de administración de alumnos				

3	Seleccionar los resultados de un alumno.		PDF, permitiendo continuar visualizando las graficas		
4	Ingresar a la lista de resultados generados del alumno por los cuestionarios que ha contestado				
5	Seleccionar "Resultados Generales" o "Resultados Detallados" dependiendo de qué manera queramos visualizar los puntajes				
6	Exportar el informe en formato PDF				

Tabla 3. 15 Caso de Prueba de Usabilidad Exportar Resultados de un Cuestionario

Escenario					
Creación y Gestión de Grupos					
ID: TC11					
Descripción:		El maestro puede crear y gestionar grupos de estudiantes.			
Requerimientos Asociados:		R17, R43, R44			
Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Ingresar al dashboard del maestro	Se muestra la interfaz con la opción de crear y gestionar grupos	Se mostró correctamente la interfaz	Aprobado.	Sin errores.
2	Acceder a la sección de gestión de grupos	Se despliega la lista de grupos existentes y la opción de crear un nuevo grupo	Al darle clic ingresó a la sección deseada.	Aprobado.	Sin errores.
3	Crear un nuevo grupo y asignarle (nombre, generación, carrera, etc.) y guardar cambios.	El grupo es creado y visible en la lista de grupos	El grupo fue de fácil creación y los alumnos pueden acceder correctamente.	Aprobado.	Sin errores.

4	Agregar alumnos al grupo manualmente o mediante códigos de acceso.	Los alumnos son añadidos correctamente al grupo	Se añadió al alumno al redactar su nombre o con un código generado.	Aprobado.	Sin errores.
5	Guardar cambios	Dar clic al botón “Añadir alumno” para poder guardar los cambios y añadir al alumno al grupo correctamente.	Se guardaron los cambios para acceder posteriormente.	Aprobado.	Sin errores.

Tabla 3. 16 Caso de Prueba de Usabilidad Creacion y Gestión de Grupos

3.10.4. Módulo de Estadísticas y Comparación de Grupos.

Escenario					
Visualización de Estadísticas del Grupo					
ID: TC12					
Descripción:		Validar que un maestro puede ver las estadísticas de su grupo en una tabla y una gráfica.			
Requerimientos Asociados:		R19, R46, R48			
Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Iniciar sesión como maestro.	Se muestra la información correctamente y sin errores de carga	Al ingresar a la sección de comparación de grupos arroja inmediatamente la tabla y graficas de comparación con los grupos seleccionados.	Aprobado.	Sin errores.
2	Acceder a la sección de estadísticas del grupo				
3	Verificar que los datos se muestren en una tabla y en una grafica				

Tabla 3. 17 Caso de Prueba de Usabilidad Estadísticas del Grupo

Escenario	
Comparación de Estadísticas Entre Grupos	
ID: TC13	
Descripción:	Validar que un maestro pueda comparar las estadísticas de dos grupos mediante una gráfica de araña
Requerimientos Asociados:	R46, R48

Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Iniciar sesión como maestro.	La comparación se muestra de forma clara y permite entender las diferencias entre los grupos	La grafica de araña se actualiza correctamente al seleccionar dos nuevos grupos lo que permite la comparación de forma clara y precisa.	Aprobado.	Sin errores.
2	Acceder a la sección de comparación de grupos				
3	Seleccionar dos grupos para comparar				
4	Verificar que la gráfica de araña se actualiza correctamente con los datos				

Tabla 3. 18 Caso de Prueba de Usabilidad Comparación Entre Grupos

3.10.5. Módulo de Reportes.

Escenario					
Generación de informes					
ID: TC014					
Descripción:		Verificar que los maestros y administradores pueden generar informes en PDF			
Requerimientos Asociados:		R21, R27, R50, R52			
Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Iniciar sesión como maestro o administrador	Los informes se generan y descargan correctamente	Se generan y descargan correctamente los informes seleccionados al presionar la opción "exportar a PDF"	Aprobado	Sin errores.
2	Acceder a cualquier resultado de autoevaluación de un alumno o de un grupo				
3	Genera un informe en PDF				

Tabla 3. 19 Caso de Prueba de Usabilidad Generación de Informes

3.10.6. Módulo de Gestión de Usuarios.

Escenario	
Gestión de Usuarios	
ID: TC15	
Descripción:	El administrador debe poder gestionar usuarios (alumnos y maestros). El maestro debe poder eliminar alumnos.
Requerimientos Asociados:	R24

Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Ingresar a la plataforma como administrados	Los usuarios deben ser creados, eliminados de manera correcta.	Se crean de manera sencilla los usuarios y funciona correctamente la opción de eliminar alumnos.	Aprobado.	Sin errores.
2	Crear el perfil de un docente				
3	Visualizar la información de un usuario existente				
4	Eliminar un usuario de la plataforma				
5	Ingresar a la plataforma como maestro				
6	Eliminar un alumno de la lista de su grupo				

Tabla 3. 20 Caso de Prueba de Usabilidad Gestión de Usuarios

3.10.7. Módulo de Perfil de Administrador.

Escenario					
Visualización de Resultados de Cuestionarios.					
ID: TC16					
Descripción:		El administrador debe poder ver los resultados de todos los cuestionarios realizados en la plataforma.			
Requerimientos Asociados:		R26			
Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Ingresar a la plataforma como administrados	Los resultados deben mostrarse correctamente en una tabla y gráficos.	Se muestra el listado de los alumnos, sin embargo, no aparece la opción de "resultados" junto con las gráficas y tablas en este apartado.	Fallido.	Colocar a la vista el botón de resultados para hacerlo más intuitivo.
2	Ir a la sección de "Administración de alumnos"				
3	En el apartado de botones de acciones por alumnos, hacer clic en "resultados"				
4	Ver una lista de todos los resultados de los cuestionarios realizados del alumno.				

Tabla 3. 21 Caso de Prueba de Usabilidad Resultados de Cuestionarios

Escenario	
Generación de Informes Detallados	

ID: TC17					
Descripción:		El administrador debe generar informes detallados de los resultados de los cuestionarios.			
Requerimientos Asociados:		R29			
Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Ingresar a la plataforma como administrador	Los informes deben contener gráficos y tablas precisas con los resultados de los cuestionarios.	No muestra la sección de gestión de grupos.	Fallido.	Ausencia de la sección de gestión de grupos.
2	Ingresar a la sección de gestión de grupos				
3	Seleccionar el grupo al cual se le generara el informe				
4	Hacer clic a "Estadísticas por grupo"				
5	Al ingresar se verá el informe estadístico de las características emprendedoras de los estudiantes de ese grupo				

Tabla 3. 22 Caso de Prueba de Usabilidad Generación de Informes Detallados

Escenario					
Consulta de Datos de Perfil de Usuarios					
ID: TC18					
Descripción:		El administrador debe poder consultar los datos de perfil de todos los usuarios			
Requerimientos Asociados:		R29			
Tarea		Resultado Esperado	Resultado Obtenido	Estado (Aprobado/Fallido)	Observaciones
1	Ingresar a la plataforma como administrados	Los datos de perfil de los usuarios deben mostrarse correctamente.	Se muestran correctamente los datos seleccionados de alumnos o maestros.	Aprobado.	Sin errores.
2	Ir a la sección de "Usuarios"				
3	Ingresar a la sección que desee consultar "Docentes" o "Alumnos"				
4	Consultar los datos de perfil de alumnos y maestros				

Tabla 3. 23 Caso de Prueba de Usabilidad Consulta de Datos de Perfil de Usuarios

3.11. PRUEBAS DE ACEPTACIÓN.

Las pruebas de aceptación se realizaron con el objetivo de verificar que el software cumple con los requisitos del cliente y está listo para su implementación. Estas pruebas validan que todas las funcionalidades clave funcionan correctamente y cumplen con las expectativas del usuario final. Estas pruebas fueron realizadas con el equipo encargado del proyecto para verificar que el sistema cumple con todo lo requerido.

3.11.1. Módulo de Bienvenida.

ID	Modulo	Objetivo de la prueba	Acción realizada	Resultado observado	Notas/Errores encontrados
PE01	Gestión de usuarios	Probar la creación de nuevos usuarios	Cree un nuevo usuario administrador	El usuario se creó correctamente	Sin errores
PE02	Asignación de cuestionarios	Intentar asignar un cuestionario sin seleccionar un grupo	Hice clic en "Publicar cuestionario" sin elegir un grupo	Mensaje de error indicando que falta un grupo	Correcto
PE03	Exportación de informes	Probar exportación de informes a PDF	Intente exportar un informe desde la sección de reportes	El archivo PDF descargado contenía información incompleta	Sin errores
PE04	Comparación de estadísticas	Evaluar la comparación entre dos grupos	Seleccione dos grupos para comparar	La grafica de araña se actualizo correctamente	Sin errores
PE05	Creación y gestión de grupos	Validar la creación de grupos con distintos parámetros	Cree un grupo con una carrera y generación asignada	El grupo se creó correctamente	Sin errores
PE06	Gestión de perfil del administrador	Verificar que el administrador pueda gestionar carreras, países e instituciones	Cree una nueva carrera y una institución desde el perfil de administrador	Los datos fueron guardados correctamente	Sin errores
PE07	Consulta de resultados de cuestionarios	Validar que el administrador y el docente pueda ver los resultados de todos los cuestionarios	Ingresé a la sección de administración de alumnos y accedí a los resultados de los alumnos	Se mostraron correctamente	Sin errores

PE08	Creación de cuestionarios	Verificar que el administrador pueda crear un cuestionario para que el docente lo publique en un grupo	Cree un cuestionario desde el perfil de administrador	El cuestionario fue guardado correctamente y esté disponible para ser publicado por el maestro	Sin errores
PE09	Navegación del menú principal	Verificar que el menú principal muestra seis botones y que cada uno dirige a la sección correspondiente	Ingresa a la página principal y revise los botones Home, Quienes somos, Contacto, Cuestionario, Registro e Inicio de sesión	Cada botón redirigió correctamente a la página correspondiente	Sin errores
PE10	Inicio de sesión	Validar que un usuario puede iniciar sesión con credenciales correctas	Ingresa correo y contraseña válidos y presione "Iniciar sesión"	El usuario fue redirigido a su perfil correctamente	Sin errores
PE11	Registro y acceso a la plataforma (Alumno)	Validar que un alumno puede registrarse y acceder correctamente	Complete el formulario de registro con datos válidos y luego inicie sesión	El alumno fue dirigido correctamente a su perfil de usuario	Sin errores
PE12	Visualización de cuestionarios asignados	Validar que el alumno puede ver sus cuestionarios pendientes	Ingresa al apartado "Cuestionarios pendientes" y seleccione un cuestionario	Se mostraron correctamente los cuestionarios pendientes	Sin errores
PE13	Responder cuestionarios	Validar que el alumno pueda completar y enviar un cuestionario	Accedí a un cuestionario, contesté las preguntas y lo envié	El cuestionario fue enviado correctamente	Sin errores
PE14	Resultados de autoevaluación	Validar que el alumno reciba retroalimentación tras completar un cuestionario	Completé un cuestionario y envié las respuestas	Se mostró correctamente la retroalimentación con puntajes y análisis	Sin errores

PE15	<i>Ver progreso individual</i>	<i>Verificar que el alumno pueda visualizar su progreso en la plataforma</i>	<i>Accedí al dashboard del estudiante y revise su progreso en cuestionarios completados</i>	<i>El progreso fue visible y preciso</i>	<i>Sin errores</i>
PE16	<i>Visualización de resultados (Macro, Intermedio y Micro)</i>	<i>Validar que el docente pueda visualizar los resultados de los alumnos en distintos niveles de detalle</i>	<i>Ingresa a la sección de administración de alumnos, accedí a los resultados de los alumnos y seleccioné a los resultados detallados</i>	<i>Se mostraron correctamente los resultados en la vista seleccionada</i>	<i>Sin errores</i>
PE17	<i>Visualización de las estadísticas grupales</i>	<i>Validar que el docente pueda ver las estadísticas detalladas de un grupo específico</i>	<i>Ingresa a la sección de grupos, para seleccionar un grupo, acceder a las estadísticas grupales</i>	<i>Se mostraron correctamente las estadísticas grupales en una tabla y su grafica</i>	<i>Sin errores</i>
PE18	<i>Comparación de grupos</i>	<i>Validar que el maestro pueda comparar estadísticas entre diferentes grupos</i>	<i>Seleccione dos grupos y revise las estadísticas en la gráfica de araña</i>	<i>Los datos se mostraron correctamente</i>	<i>Sin errores</i>
PE19	<i>Comparación entre estudiantes</i>	<i>Validar que el docente pueda comparar el desempeño entre estudiantes individuales</i>	<i>Ingresa al apartado de comparación entre estudiantes, seleccione dos grupos, para poder seleccionar un estudiante por cada grupo y el cuestionario, y revise los datos en la vista de comparación</i>	<i>La información y grafica de araña con la comparación se presentó correctamente</i>	<i>Sin errores</i>

Tabla 3. 24 Casos de Pruebas de Aceptación

Capítulo 4. RESULTADOS Y DISCUSION.

En este capítulo, se presentan los resultados obtenidos durante la implementación del sistema de inteligencia de negocios para la identificación de patrones en los perfiles emprendedores de los egresados universitarios específicamente en la carrera de **Ingeniería en Innovación Agrícola Sustentable** del **ITESCAM**, el cual fue previamente seleccionado como el principal marco de referencia para la obtención de resultados para esta prueba. El objetivo principal de esta evaluación fue analizar como las características emprendedoras de los egresados se alinean con el perfil de egreso establecido para la carrera. Como primer paso para la realización de este análisis se realizó la comparación de las 10 características emprendedoras evaluadas en el cuestionario (**CEPs**) con el perfil de egreso de la carrera. Este perfil establece las competencias y habilidades que un egresado de la carrera debe poseer al finalizar su formación, enfocándose en áreas clave como el diseño, la innovación, la gestión de empresas agrícolas y la promoción de prácticas sustentables.

A continuación, se presentan los resultados de la comparación, donde se destacan las características emprendedoras que se alinean con los objetivos y capacidades descritas en el perfil de egreso, así como las áreas donde ocurre una colaboración eficaz y clara entre los atributos del emprendimiento y las competencias específicas del egresado.

Perfil de Egreso de la carrera Ingeniería en Innovación Agrícola Sustentable	Características emprendedoras del cuestionario (CEPs)
❖ Diseñar, crear, instalar, operar y proporcionar mantenimiento a empresas agrícolas dentro de un marco de desarrollo regional, nacional e internacional. ❖ Desarrollar, coordinar, asesorar y dirigir empresas basadas en sistemas de producción de agricultura protegida, considerando las adecuaciones que implican las condiciones ecológicas y socioeconómicas de las distintas regiones del país. ❖ Aplicar conocimientos técnicos, administrativos, culturales y de extensión para el uso eficiente del agua en la agricultura mediante la implementación y manejo adecuado de los sistemas de riego, con base en las necesidades hídricas de los cultivos, condiciones climáticas, edáficas y sociales. ❖ Promover la organización de los productores y asesorar a éstos en los	a) ATRIBUTO EMPRENDEDOR • Buscar y aprovechar las oportunidades: Los emprendedores mexicanos destacan por su capacidad para identificar necesidades insatisfechas en el mercado y desarrollar soluciones innovadoras. Esta habilidad les permite crear propuestas de valor únicas y competitivas. (Blog, 2023) • Fijar objetivos: Establecer metas claras y alcanzables es fundamental para orientar los esfuerzos y recursos de manera efectiva. Los emprendedores exitosos en México suelen definir objetivos específicos que guían el crecimiento de sus negocios. (Empleo, s.f.) • Planificar y hacer seguimiento sistemático: La planificación estratégica y el monitoreo constante del progreso permiten a los emprendedores adaptarse a cambios y optimizar sus operaciones. Esta práctica es común entre los emprendedores mexicanos que buscan sostenibilidad a largo plazo (Laura Iturbide Galindo, 2020)

sistemas de agricultura protegida en cultivos estratégicos; así como garantizar una producción con calidad y proponer innovaciones tecnológicas. ❖ Participar con organizaciones, grupos de trabajo y productores interesados en el establecimiento de proyectos productivos sobre agricultura protegida e impulsar la aplicación de buenas prácticas agrícolas y de manufactura. ❖ Realizar labores profesionales de docencia, investigación y extensión de nuevos conocimientos, metodologías y tecnologías, en los ámbitos de la agricultura protegida, inocuidad alimentaria y bioseguridad, para el mejor aprovechamiento de los recursos del agro. ❖ Innovar los sistemas de producción agrícola a través de la aplicación de modelos de predicción mediante el uso de herramientas informáticas, para la toma de decisiones en las actividades de riego, nutrición, manejo integrado de plagas, enfermedades y riesgo climático. ❖ Solucionar problemas en la producción, manejo y comercialización de productos agrícolas, con el empleo racional e innovador de los recursos naturales y materiales en la producción agrícola. ❖ Participar en la solución de la problemática del sector agrícola y hacer sustentable el uso de los recursos naturales en su producción, transformación y comercialización en el ámbito nacional e internacional. ❖ Generar, validar y adoptar nuevas tecnologías compatibles con la situación socioeconómica del entorno regional para aprovechar el potencial productivo de las áreas agrícolas en función de sus condiciones ecológicas. ❖ Analizar la problemática agrícola en términos tecnológicos, ecológicos, económicos y sociales, para establecer las bases de un manejo sustentable	b) ACTITUD EMPRENDEDORA • Ser persistentes: Como emprendedores, tener persistencia es fundamental para poder alcanzar el éxito, porque nos permite levantarnos y seguir adelante a pesar de los obstáculos. Esta actitud nos permite superar las adversidades y mantener el enfoque en nuestros objetivos. (Blanco, 2021) • Cumplimiento de compromisos: Es una cualidad importante y virtuosa, ya que puede asegurar el éxito en proyectos futuros. Los emprendedores deben de honrar recuerdos con sus clientes, proveedores y socios esto es esencial para construir una reputación sólida y relaciones de confianza. (Cavero, Cumplimiento de compromisos, 2023) • Exigir eficiencia y calidad: En un emprendimiento significa que debemos de buscar hacer las cosas de manera más rápida, mejor y más económica. Esto implica cumplir las normas de calidad y excelencia. Como emprendedores debemos de buscar constantemente mejorar la calidad de nuestros productos o servicios y optimizar procesos para satisfacer las expectativas del mercado. (Indeed, 2024) c) CAPACIDAD EMPRENDEDORA • Correr riesgos moderados: Esta cualidad implica que el emprendedor debe de evaluar cuidadosamente las posibles consecuencias antes de tomar decisiones que puedan ocasionar un riesgo. No se trata de evitar el riesgo por completo, pero si debemos de asumir aquellos que son calculados y que nos ofrezcan una relación favorable entre el riesgo y beneficio. (Pedraza, 2023) • Tener autoconfianza: Esta característica se refiere a la seguridad que el emprendedor debe de sentir en sus propias habilidades y juicios. Esta confianza y seguridad que sienten les permite enfrentar desafíos y
---	--

basado en las formas de organización social predominantes en la región. ❖ Manejar y aplicar metodologías y técnicas que permitan obtener información pertinente para planear actividades de un sistema productivo sustentable. ❖ Desarrollar sistemas de producción agrícola sustentable, a través del uso racional y organizado de los recursos regionales disponibles y la innovación tecnológica con la finalidad de mejorar su productividad y lograr un equilibrio ambiental. ❖ Conocer y aplicar la legislación nacional e internacional, en la producción, transformación y comercialización de productos y servicios agrícolas con estándares de calidad.	tomar las decisiones con determinación, incluso en las situaciones de incertidumbre. (Pedraza, 2023) d) CAPACIDAD INTERPERSONAL • Buscar y conseguir información: Esta habilidad consiste en la proactividad del emprendedor para obtener información relevante que les ayude en la toma de decisiones. Los emprendedores necesitan estar al tanto de las últimas tendencias, oportunidades del mercado y las necesidades que tiene sus clientes para poder crear sus productos o servicios que satisfagan estas necesidades y que al mismo tiempo sean exitosas en el mercado. (Cavero, Búsqueda de información. 3ª Competencia emprendedora, 2023) • Persuasión y creación de redes de apoyo: Esta capacidad en un emprendedor significa influir en otras personas y establecer una red de contactos que le puedan brindar apoyo en los diferentes aspectos de su emprendimiento. Es decir que deben de tener la habilidad para negociar, convencer y mantener relaciones beneficiosas con los clientes, proveedores e inversores. (Bernal, 2024)
--	--

Tabla 4. 1 Comparación Entre Perfil de Egreso y Características Emprendedoras

A continuación, se presentan los resultados de la comparación:

a) ATRIBUTO EMPRENDEDOR

- **Buscar y aprovechar las oportunidades:** Esta característica nos muestra una fuerte alineación con la habilidad de **diseñar, crear, instalar y operar empresas agrícolas** en diversos contextos, nos refleja la capacidad del egresado para identificar y aprovechar oportunidades dentro del marco de desarrollo regional y nacional.
- **Fijar objetivos:** La capacidad de **definir metas claras y alcanzables** se ve reflejada en la habilidad del egresado para **desarrollar y coordinar empresas** que estén basadas en sistemas agrícolas protegidas, lo que requiere la planificación de sus metas a corto y largo plazo de acuerdo con sus condiciones locales.
- **Planificar y hacer seguimiento sistemático:** El egresado debe de aplicar **conocimientos técnicos y administrativos** para poder implementar y gestionar sistemas de riego de manera eficiente, lo que implica que debe de tener la capacidad de planificar y hacer el seguimiento de proyectos a largo plazo.

b) ACTITUD EMPRENDEDORA

- **Ser persistentes:** La persistencia es fundamental para **promover la organización de los productores y asesorarlos en la agricultura protegida**, lo cual requiere un compromiso constante para superar los desafíos que puedan surgir en la implementación de proyectos agrícolas.
- **Cumplimiento de compromisos:** Como egresado debe demostrar su capacidad para **realizar labores profesionales de docencia, investigación y extensión**, lo que implica que debe de cumplir con los compromisos adquiridos con personas importantes en el sector agrícola.
- **Exigir eficiencia y calidad:** La innovación de los sistemas de producción agrícola, mediante el uso de herramientas informáticas para **modelos de predicción**, resalta la importancia de optimizar procesos y asegurar la **calidad y eficiencia** en la producción agrícola.

c) CAPACIDAD EMPRENDEDORA

- **Correr riesgos moderados:** En la solución de problemas agrícolas, el egresado debe estar dispuesto a **asumir riesgos moderados**, evaluando las mejores estrategias para la producción, manejo y la comercialización de productos agrícolas, sin comprometer los recursos naturales con los que cuenta.
- **Tener autoconfianza:** Esta característica está vinculada con la capacidad del egresado para **participar en la solución de problemas del sector agrícola**, enfrentando así retos de manera proactiva y con una fuerte seguridad en sus decisiones.

d) CAPACIDAD INTERPERSONAL

- **Buscar y conseguir información:** Como egresado debe ser capaz de **generar, validar y adoptar nuevas tecnologías**, lo cual implica una constante búsqueda de información y actualización en el ámbito agrícola para promover soluciones innovadoras.
- **Persuasión y creación de redes de apoyo:** La capacidad de **analizar la problemática agrícola** y proponer soluciones sustentables requieren de habilidades interpersonales para persuadir y crear redes de apoyo, tanto dentro de la comunidad agrícola como con otras personas que son importante en el sector agrícola.

Una vez obtenidos los resultados de la comparación entre las características emprendedoras y el perfil de egreso, procedemos a llevar a cabo la segunda fase para la obtención de resultados que es la aplicación de la prueba a los estudiantes de primero y octavo semestre de la carrera. Esto nos permitirá analizar en qué nivel de desarrollo de sus competencias emprendedoras (CEPs) y así poder evaluar que tan alineadas están con su perfil de egreso. A partir de esta comparación que se realizará, nos será posible determinar la evolución de dichas competencias a lo largo de la carrera y así obtener información clave para la conclusión de nuestro estudio.

Cuando el alumno acceda a su perfil de usuario, podrá visualizar sus cuestionarios que tiene pendiente, podrá iniciar ese cuestionario para poder responder las preguntas e inmediatamente recibir los resultados.

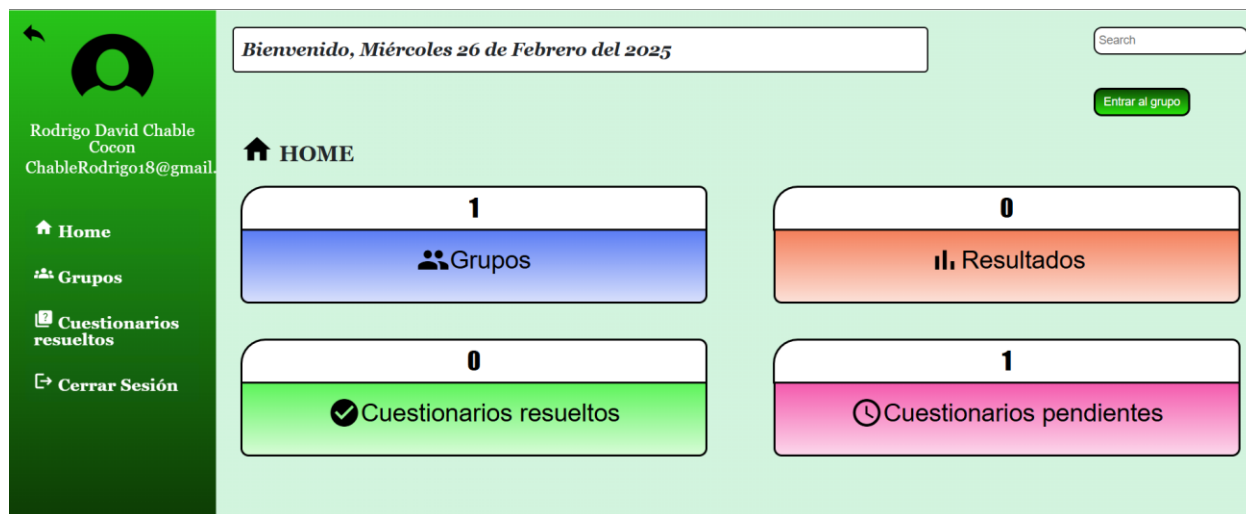


Ilustración 4. 1 Dashboard de los Alumnos

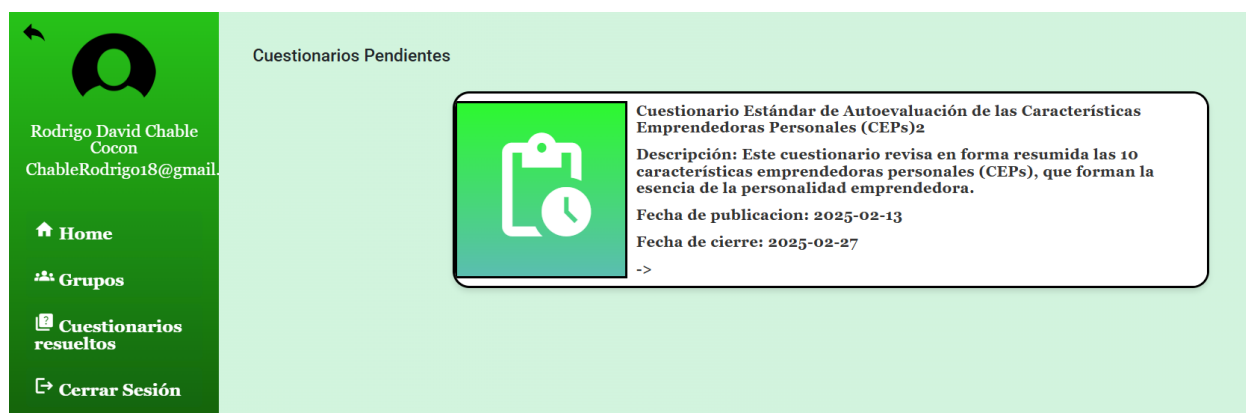


Ilustración 4. 2 Lista de Cuestionarios Pendientes



Ilustración 4. 3 Vista de Instrucciones del Cuestionario

Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs)2

Este cuestionario revisa en forma resumida las 10 características emprendedoras personales (CEPs), que forman la esencia de la personalidad emprendedora.

Me esmero en buscar cosas que necesitan hacerse.

☐ Nunca
☐ Raras veces
☐ Algunas veces
☐ Usualmente
☐ Siempre

Cuando me enfrento a un problema difícil invierto gran cantidad de tiempo en encontrar una solución.

☐ Nunca
☐ Raras veces
☐ Algunas veces
☐ Usualmente
☐ Siempre

Ilustración 4. 4 Vista del Cuestionario

☐ Siempre

Puedo lograr que personas con firmes convicciones y opiniones cambien de modo de pensar.

☐ Nunca
☐ Raras veces
☐ Algunas veces
☐ Usualmente
☐ Siempre

Me mantengo firme en mis decisiones, aún cuando otras personas me contradigan enérgicamente.

☐ Nunca
☐ Raras veces
☐ Algunas veces
☐ Usualmente
☐ Siempre

Cuando no se algo no reparo en admitirlo.

☐ Nunca
☐ Raras veces
☐ Algunas veces
☐ Usualmente
☐ Siempre

Anterior Enviar

Ilustración 4. 5 Vista del Cuestionario (Preguntas y Respuestas)

Al finalizar el cuestionario, el sistema mostrara la siguiente pantalla con los siguientes detalles: los datos importantes del alumno, junto con los del cuestionario (nombre, descripción y fecha de publicación).



Rodrigo David Chable Cocon
ChableRodrigo18@gmail.com

- Home
- Grupos
- Cuestionarios resueltos
- Cerrar Sesión

Exportar a PDF

Resultados de Rodrigo David Chable Cocon (7018)

Nombre: Rodrigo David Chable Cocon
Correo: ChableRodrigo18@gmail.com
Semestre: 2º
Carrera: INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE
Nombre del cuestionario: Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs)2
Descripción: Este cuestionario revisa en forma resumida las 10 características emprendedoras personales (CEPs), que forman la esencia de la personalidad emprendedora.
Fecha de publicación:

Resultado de autoevaluación

Variable	Puntaje
Factor de corrección	18
Buscar y aprovechar las oportunidades	15
Ser Persistentes	17

Ilustración 4. 6 Información importante del Alumno y Cuestionario (Resultados)



Rodrigo David Chable Cocon
ChableRodrigo18@gmail.com

- Home
- Grupos
- Cuestionarios resueltos
- Cerrar Sesión

Resultado de autoevaluación

Variable	Puntaje
Factor de corrección	18
Buscar y aprovechar las oportunidades	15
Ser Persistentes	17
Cumplimiento de compromisos	16
Exigir eficiencia y calidad	16
Correr riesgos moderados	19
Fijar Objetivos	14
Buscar y conseguir información	16
Planificar y hacer seguimiento sistemático	12
Persuasión y creación de redes de apoyo	19
Tener autoconfianza	15

Gráfico de autoevaluación

 Puntajes por variable

Ilustración 4. 7 Vista de la tabla de puntajes obtenidos

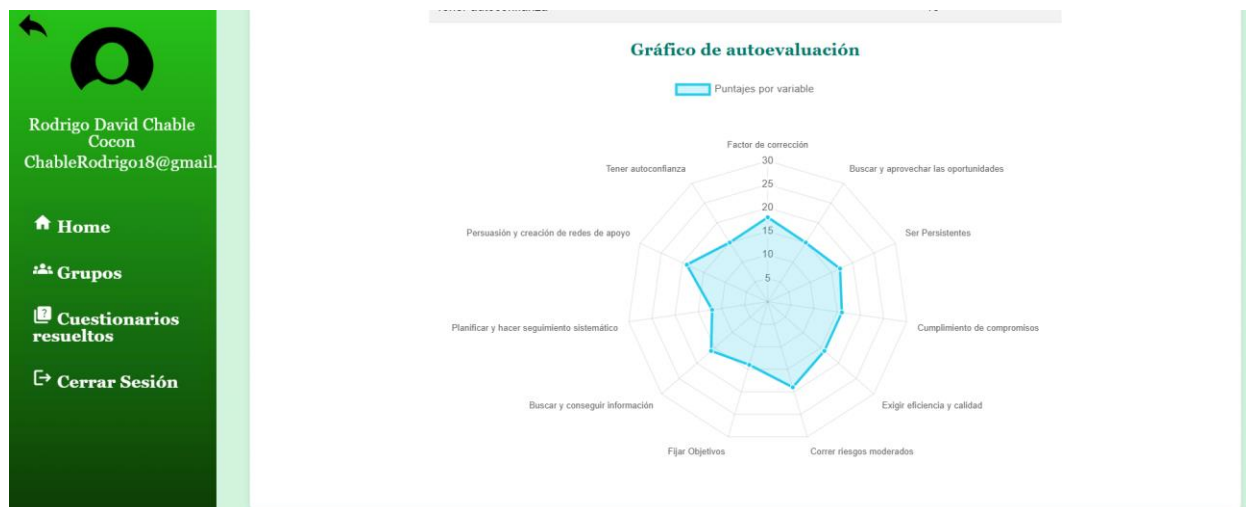


Ilustración 4. 8 Grafica araña de los puntajes obtenidos

Estos resultados de los puntajes de cada característica son esenciales, pero para iniciar el análisis de estos, nos enfocaremos en la plataforma del docente quien es, el que monitoreará los perfiles de los estudiantes, aquí se realizara la comparación entre los grupos que tiene creados y los estudiantes que tenga inscritos en estos, esto nos ayudara a llevar a cabo el estudio

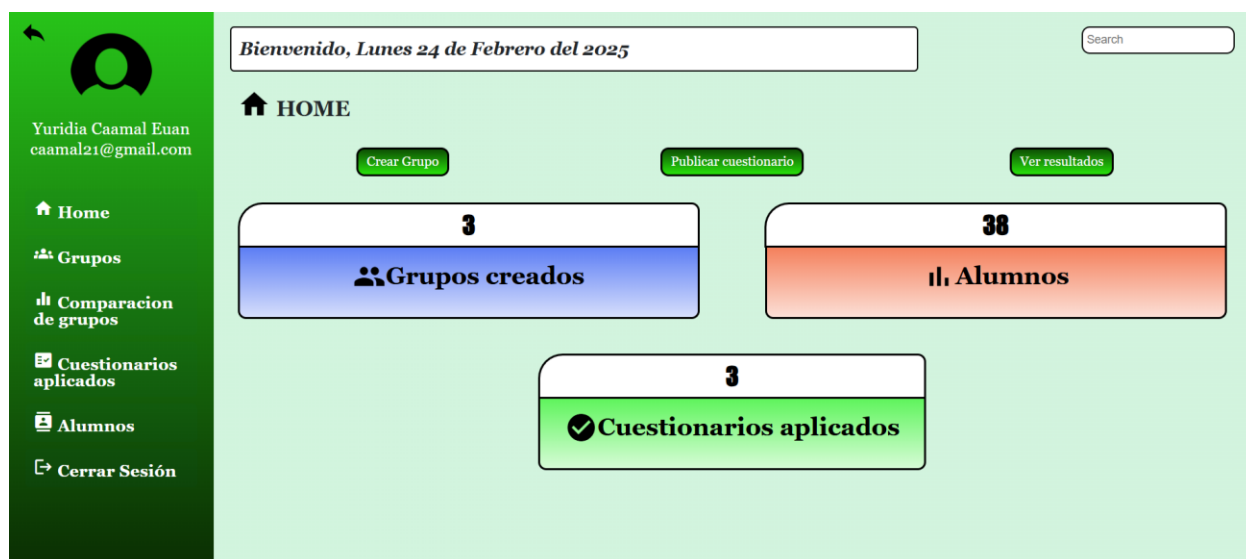


Ilustración 4. 9 Dashboard del Docente

Al ingresar a la plataforma nos dirigiremos al apartado de “**Grupos creados**”, nos desplegara la sección en donde podremos visualizar los grupos que tenemos creados. En este caso nuestro docente solo tiene dos grupos como parte de la prueba un **grupo de primer semestre** y otro de **octavo semestre** de la carrera **Ingeniería en Innovación Agrícola Sustentable**



Ilustración 4. 10 Visualización de los Grupos del Docente

Seleccionaremos el grupo del primer semestre, al ingresar a la sección podremos visualizar los datos importantes de este grupo como (institución, carrera, semestre, nombre del grupo, código de acceso) así como la visualización de los cuestionarios que ya han sido aplicados a este grupo, como podremos notar podemos entrar para visualizar las estadísticas del cuestionario que ya fue aplicado a ese grupo y poder ver los datos de los integrantes que están inscritos.



Ilustración 4. 11 Vista de la Información del Grupo y Cuestionarios Aplicados

Al seleccionar el botón de “Ver integrantes”, nos mandará a la sección en donde se presentará una tabla con la información de los estudiantes inscritos como (Nombre, Primer apellido, Segundo apellido, Carrera, Institución, Correo, Semestre) los botones que se encuentran en la última columna de “Acciones” son  para visualizar los puntajes de los cuestionarios contestados de ese alumno en específico y el otro botón  es para eliminar al alumno del grupo si así lo deseamos.

Los botones que se encuentran arriba de la tabla son “Buscar y añadir alumno” que sirve para cuando el docente quiera añadir el mismo a los alumnos le sea más fácil buscarlo entre todos los alumnos que hay inscritos en la plataforma para añadirlo a su grupo y pueda resolver los cuestionarios y monitorear sus resultados. El otro botón “Exportar PDF” es para poder exportar esta misma tabla a un archivo PDF cuando el docente lo necesite.

Administración de Alumnos							
BUSCAR Y AÑADIR ALUMNO		Exportar a PDF					
NOMBRE	PRIMER APELLIDO	SEGUNDO APELLIDO	CARRERA	INSTITUCION	CORREO	SEMESTRE	ACCIONES
Cinthia Araceli	Chi	Poot	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	ChiCinthia69@gmail.com	1*	 
Rubi Guadalupe	Chable	Uc	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	rubChable58@gmail.com	1*	 
Yareni Guadalupe	Canul	May	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	yareniCanul63@gmail.com	1*	 
Mayte Jael	Naal	Ucan	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	MayteNaal64@gmail.com	1*	 
Sarahi Guadalupe	Bacab	Haaz	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	BacabSarahi68@gmail.com	1*	 
Roxana Itzel	Euan	Tuz	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	EuanRoxana59@gmail.com	1*	 
Pedro Alvarado	Xaman	Dzib	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	XamanPedro45@gmail.com	1*	 
Sheldren Evelin	Tun	Ek	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	TunSheldren59@gmail.com	1*	 
Jorge Carlos	Valle	Haas	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	ValleJorge47@gmail.com	1*	 
Elly Noeli	Kuk	Euan	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	KukElly48@gmail.com	1*	 
Shermi Noemi	Centeno	Ucan	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	CentenoShermi78@gmail.com	1*	 

Ilustración 4. 12 Tabla con la información de los alumnos del grupo seleccionado

Administración de Alumnos					
BUSCAR Y AÑADIR ALUMNO		Exportar a PDF			
NOMBRE	PRIMER APELLIDO	SEGUNDO APELLIDO	CARRERA	INSTITUCION	
Cinthia Araceli	Chi	Poot	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	
Rubi Guadalupe	Chable	Uc	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	
Yareni Guadalupe	Canul	May	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	
Mayte Jael	Naal	Ucan	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	
Sarahi Guadalupe	Bacab	Haaz	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	
Roxana Itzel	Euan	Tuz	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	
Pedro Alvarado	Xaman	Dzib	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	
Sheldren Evelin	Tun	Ek	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	

Ilustración 4. 13 Tabla con la información de los alumnos del grupo seleccionado

Administración de Alumnos						
 Yuridia Caamal Euan caamal21@gmail.com Home Grupos Comparacion de grupos Cuestionarios aplicados Alumnos Cerrar Sesión	Exportar a PDF					
	ID	SEGUNDO APELLIDO	CARRERA	INSTITUCION	CORREO	SEMESTRE
		Poot	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	ChiCinthia69@gmail.com	1*
		Uc	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	rubiChable58@gmail.com	1*
		May	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	yarenCanul63@gmail.com	1*
		Ucan	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	MayteNaal64@gmail.com	1*
		Haaz	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	BacabSarahi68@gmail.com	1*
		Tuz	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	EuanRoxana59@gmail.com	1*
		Dzib	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	XamanPedro45@gmail.com	1*
		Ek	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	TunSheldren59@gmail.com	1*
ACCIONES						
             						

Ilustración 4. 14 Tabla con la información de los alumnos del grupo seleccionado (Botones de acciones)

Al acceder a la sección de resultados del alumno que hemos seleccionado, se muestra una lista con todos los cuestionarios que han respondido hasta el momento. Desde esta sección, nos es posible visualizar los resultados en distintos niveles de detalle. Podemos optar por una **vista general**, que nos presenta las puntuaciones en un solo formato, como lo visualiza en su momento el alumno, o podemos optar por una **vista detalla** que trae la **vista Macro** que nos presenta las puntuaciones por categoría, una **vista intermedia**, que nos desglosa las puntuaciones de cada expresión dentro de las categorías, o una **vista Micro** que nos muestra el puntaje obtenido en cada pregunta que conforma cada expresión.

Cuestionarios Resueltos	
Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs)2	
Ver Resultados	Ver Resultados Generales

Ilustración 4. 15 Lista de los cuestionarios resueltos por alumno

Al ingresar al apartado de los **resultados generales**, se muestra los datos importantes del alumno, así como del cuestionario que se contestó, de igual manera se visualiza una tabla con los puntajes obtenidos en el cuestionario, desglosados por variables de las características emprendedoras. Además, se presenta una gráfica de araña que permite visualizar de forma clara y comparativa el

desempeño del alumno en la relación con los diferentes parámetros evaluados, facilitando así una interpretación rápida y efectiva de sus resultados.



Resultados de Cinthia Araceli Chi Poot (8769)

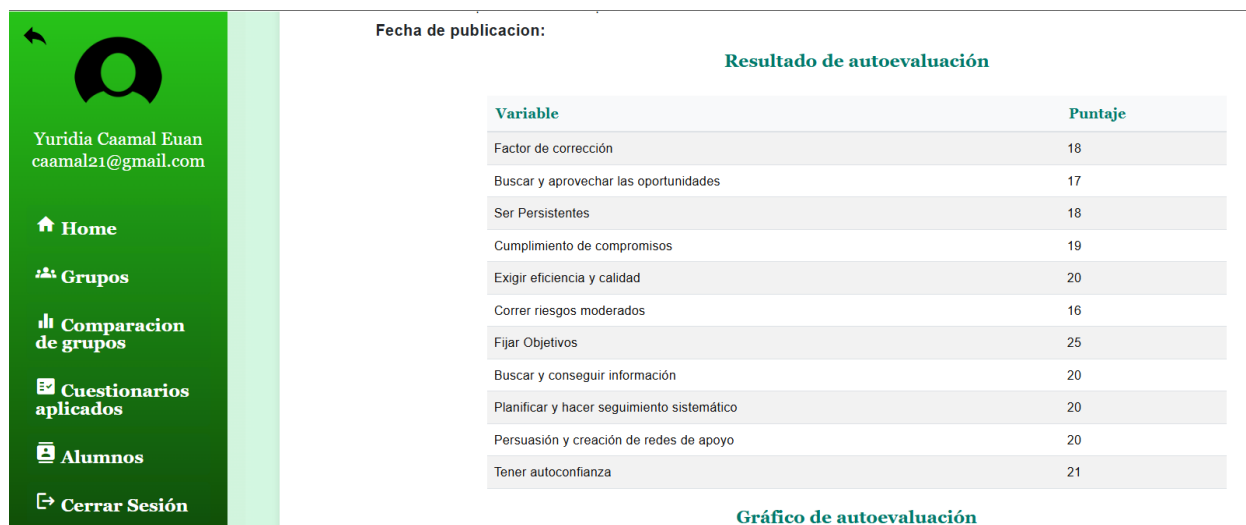
Nombre: Cinthia Araceli Chi Poot
Correo: ChiCinthia69@gmail.com
Semestre: 1°
Carrera: INGENIERIA EN INNOVACIÓN AGRÍCOLA SUSTENTABLE

Nombre del cuestionario: Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs)²
Descripción: Este cuestionario revisa en forma resumida las 10 características emprendedoras personales (CEPs), que forman la esencia de la personalidad emprendedora.
Fecha de publicación:

Resultado de autoevaluación

Variable	Puntaje
Factor de corrección	18
Buscar y aprovechar las oportunidades	17
Ser Persistentes	18
Cumplimiento de compromisos	19

Ilustración 4. 16 Vista de Puntajes Obtenidos (Información del alumno y cuestionario)



Fecha de publicación:

Resultado de autoevaluación

Variable	Puntaje
Factor de corrección	18
Buscar y aprovechar las oportunidades	17
Ser Persistentes	18
Cumplimiento de compromisos	19
Exigir eficiencia y calidad	20
Correr riesgos moderados	16
Fijar Objetivos	25
Buscar y conseguir información	20
Planificar y hacer seguimiento sistemático	20
Persuasión y creación de redes de apoyo	20
Tener autoconfianza	21

Gráfico de autoevaluación

Ilustración 4. 17 Vista de Puntajes Obtenidos (Tabla de puntajes en las características)



Ilustración 4. 18 Vista de Puntajes Obtenidos (Grafica de araña)

Gráfico de autoevaluación

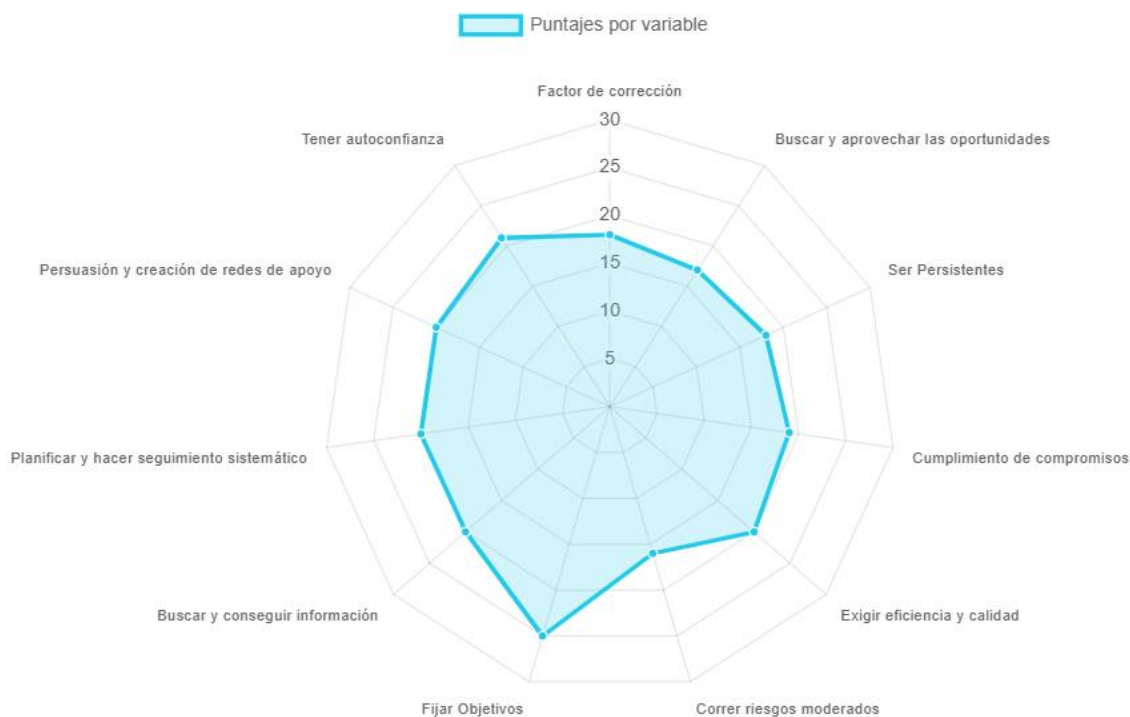


Ilustración 4. 19 Vista de Puntajes Obtenidos en las características (Grafica de araña)

Por otro lado, al ingresar al apartado de ver resultados de manera detallada, se presenta un desglose de los puntajes obtenidos por cada alumno en las diferentes categorías evaluadas dentro del cuestionario. En esta vista, se pueden analizar los resultados desde tres niveles:

- **Vista Macro:** Muestra un resumen general de los puntajes obtenidos en cada una de las categorías evaluadas con el cuestionario, tales como “**Atributo Emprendedor**” y “**Actitud emprendedora**”. Se presentan mediante barras de progreso que indican el puntaje alcanzado en la relación con el puntaje máximo posible, con esto nos facilita una interpretación rápida del desempeño global del alumno.

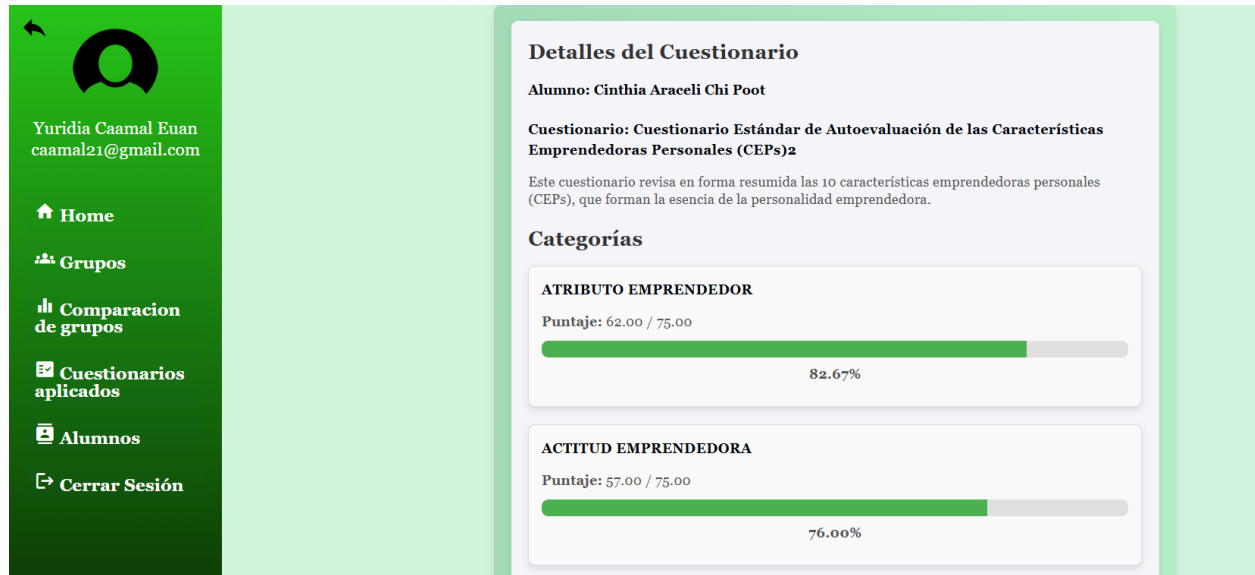


Ilustración 4. 20 Vista Macro de puntajes por categoría

Detalles del Cuestionario

Alumno: Cinthia Araceli Chi Poot

Cuestionario: Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs)²

Este cuestionario revisa en forma resumida las 10 características emprendedoras personales (CEPs), que forman la esencia de la personalidad emprendedora.

Categorías

ATRIBUTO EMPRENDEDOR

Puntaje: 62.00 / 75.00



82.67%

ACTITUD EMPRENDEDORA

Puntaje: 57.00 / 75.00



76.00%

Ilustración 4. 21 Vista Macro características que conforman el cuestionario



Ilustración 4. 22 Vista Macro características que conforman el cuestionario

- **Vista intermedia:** Esta sección nos permite analizar con mayor profundidad los puntajes obtenidos en cada una de las expresiones que conforman las categorías. esta vista nos proporciona un nivel más detallado de la información, nos ayuda a identificar las fortalezas y áreas de mejora en cada aspecto evaluado.

Detalles del Cuestionario

Alumno: Cinthia Araceli Chi Poot

Cuestionario: Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs)²

Este cuestionario revisa en forma resumida las 10 características emprendedoras personales (CEPs), que forman la esencia de la personalidad emprendedora.

Expresiones - Categoría: ATRIBUTO EMPRENDEDOR

	Buscar y aprovechar las oportunidades	Puntaje: 17.00 / 25.00	Ver Preguntas
	Fijar Objetivos	Puntaje: 25.00 / 25.00	Ver Preguntas
	Planificar y hacer seguimiento sistemático	Puntaje: 20.00 / 25.00	Ver Preguntas



Ilustración 4. 23 Vista Intermedio de las Expresiones

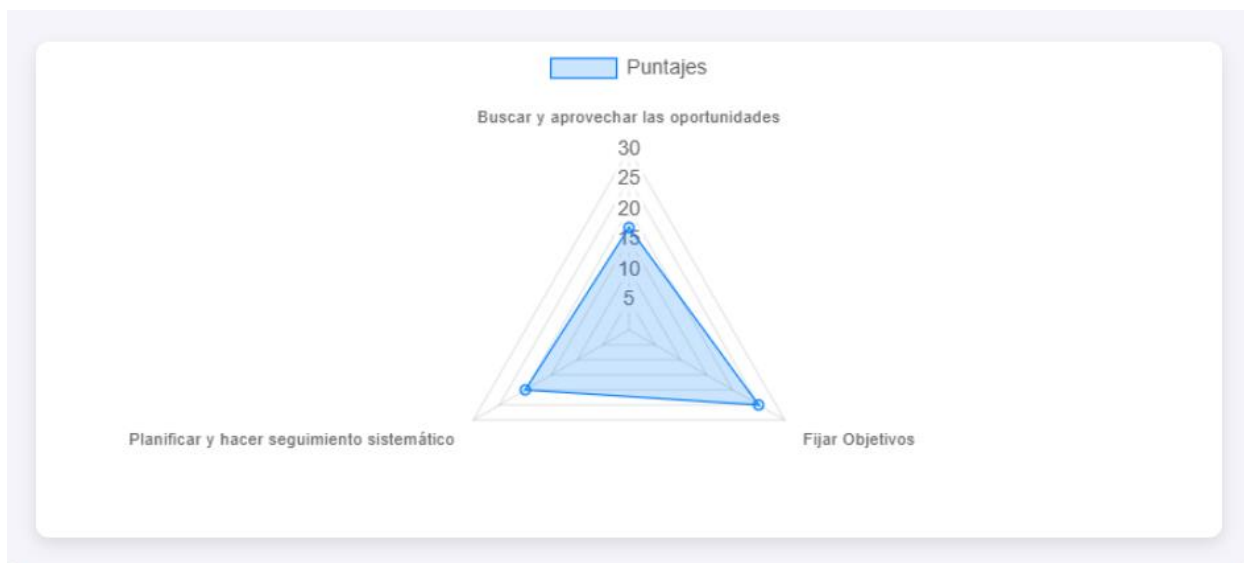


Ilustración 4. 24 Vista Intermedia de la gráfica de araña de las expresiones

- **Vista Micro:** Nos presenta un análisis a nivel de pregunta, mostrando los puntajes obtenidos en cada ítem del cuestionario. Esta vista nos ofrece la mayor precisión posible, ya que permite observar el desempeño del alumno en cada una de las preguntas que conforman las expresiones y categorías del cuestionario.

Yuridia Caamal Euan
caamal21@gmail.com

- Home
- Grupos
- Comparacion de grupos
- Cuestionarios aplicados
- Alumnos
- Cerrar Sesión

Detalles del Cuestionario

Alumno: Cinthia Araceli Chi Poot

Cuestionario: Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs)²

Este cuestionario revisa en forma resumida las 10 características emprendedoras personales (CEPs), que forman la esencia de la personalidad emprendedora.

Preguntas - Expresión: Buscar y aprovechar las oportunidades

Pregunta	Respuesta del alumno	Puntaje máximo
Me esmero en buscar cosas que necesitan hacerse.	1	5
Hago lo que se necesita hacer sin que otros tengan que pedirme que lo haga.	5	5
Me gustan los desafíos y nuevas oportunidades.	5	5
Prefiero desempeñar tareas que domino a la perfección y en las que me siento seguro.	5	5
Me aventura a hacer cosas nuevas y diferentes de lo que he realizado en el pasado.	5	5

Ilustración 4. 25 Vista Micro de las respuestas del alumno por pregunta

Gracias a esta estructura en la visualización de los resultados, es posible obtener una comprensión integral del perfil emprendedor del alumno y su evolución a lo largo del tiempo esto nos facilita la identificación de los patrones y tendencias en su desarrollo emprendedor

Dentro del sistema igual se encuentra una sección que les será de mucha ayuda a los docentes y administradores. Nos permite visualizar y analizar las estadísticas grupales obtenidas a partir

de los cuestionarios aplicados a los estudiantes. Al ingresar, se despliega una tabla con los resultados agregados del grupo seleccionado, nos muestra las métricas clave de cada categoría y variable evaluada en el cuestionario, como el promedio, la desviación estándar, los valores mínimo y máximo, la mediana y los cuartiles. Esta información nos permitirá obtener una visión general del desempeño del grupo en cada una de las dimensiones evaluadas.

GRUPO: GRUPO 1 "A"								
Cuestionario: Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs) ²								
Fecha de Publicación:								
Categoría	Variable	Promedio	Desviación Estándar	Mínimo	Máximo	Mediana	Cuartil 1	Cuartil 3
ATRIBUTO EMPRENDEDOR	Objetivos	19.04347826087	3.0072376462244	11	25	19	18.5	20
ATRIBUTO EMPRENDEDOR	Iniciativa	16.739130434783	2.614946397927	12	22	17	15	18
ATRIBUTO EMPRENDEDOR	Planificación	16.652173913043	2.0803997791547	13	20	17	15	18.5
ACTITUD EMPRENDEDORA	Persistente	16.869565217391	2.3799226081016	12	21	17	15	18.5
ACTITUD EMPRENDEDORA	Eficiencia	16.217391304348	2.4486827950615	12	20	17	14	18
ACTITUD EMPRENDEDORA	Cumplimiento	17.913043478261	2.6096506815744	14	24	18	16	20
CAPACIDAD EMPRENDEDORA	Arriesgado	16.173913043478	3.1137842208337	10	22	16	15	18
CAPACIDAD EMPRENDEDORA	Autoconfianza	17.086956521739	2.5210575608852	11	21	17	16	18.5
CAPACIDAD INTERPERSONAL	Persuasión	16.521739130435	2.0641873861686	13	21	16	15	18
CAPACIDAD INTERPERSONAL	Buscar	17.391304347826	2.2102885544813	13	22	18	16	19
FACTOR DE CORRECCION	Factor de corrección	18.695652173913	1.8934044040178	15	22	18	17	20

Ilustración 4. 26 Vista de la tabla de las Estadísticas Grupales

GRUPO: GRUPO 1 "A"								
Cuestionario: Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs) ²								
Fecha de Publicación:								
Categoría	Variable	Promedio	Desviación Estándar	Mínimo	Máximo	Mediana	Cuartil 1	Cuartil 3
ATRIBUTO EMPRENDEDOR	Objetivos	19.04347826087	3.0072376462244	11	25	19	18.5	20
ATRIBUTO EMPRENDEDOR	Iniciativa	16.739130434783	2.614946397927	12	22	17	15	18
ATRIBUTO EMPRENDEDOR	Planificación	16.652173913043	2.0803997791547	13	20	17	15	18.5
ACTITUD EMPRENDEDORA	Persistente	16.869565217391	2.3799226081016	12	21	17	15	18.5
ACTITUD EMPRENDEDORA	Eficiencia	16.217391304348	2.4486827950615	12	20	17	14	18
ACTITUD EMPRENDEDORA	Cumplimiento	17.913043478261	2.6096506815744	14	24	18	16	20
CAPACIDAD EMPRENDEDORA	Arriesgado	16.173913043478	3.1137842208337	10	22	16	15	18
CAPACIDAD EMPRENDEDORA	Autoconfianza	17.086956521739	2.5210575608852	11	21	17	16	18.5
CAPACIDAD INTERPERSONAL	Persuasión	16.521739130435	2.0641873861686	13	21	16	15	18
CAPACIDAD INTERPERSONAL	Buscar	17.391304347826	2.2102885544813	13	22	18	16	19
FACTOR DE CORRECCION	Factor de corrección	18.695652173913	1.8934044040178	15	22	18	17	20

Ilustración 4. 27 Vista de la tabla de las Estadísticas Grupales

Además de esta tabla de datos, el sistema incorpora una herramienta de visualización gráfica, que nos permite seleccionar una categoría específica, así como elegir la métrica de interés. Esta grafica de barras nos facilita la interpretación de los datos al presente de manera clara la distribución de los puntajes en cada variable dentro de la categoría seleccionada. Este análisis

nos será de mucha utilidad en el estudio ya que nos permite identificar patrones de desempeño en el grupo, así como comparar niveles de desarrollo en diferentes áreas.

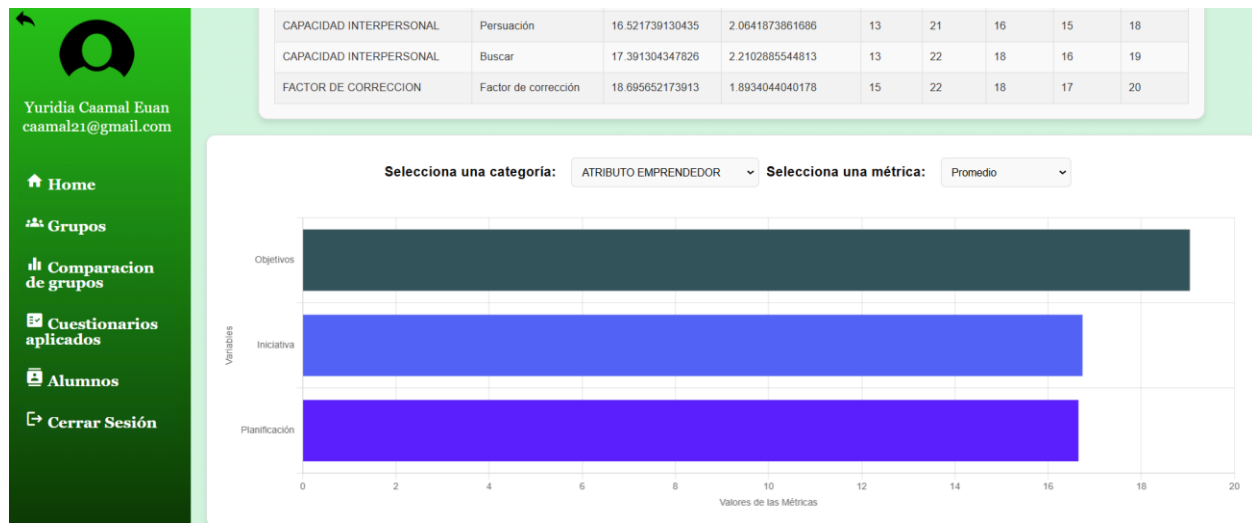


Ilustración 4. 28 Grafico de comparación de las Estadísticas Grupales

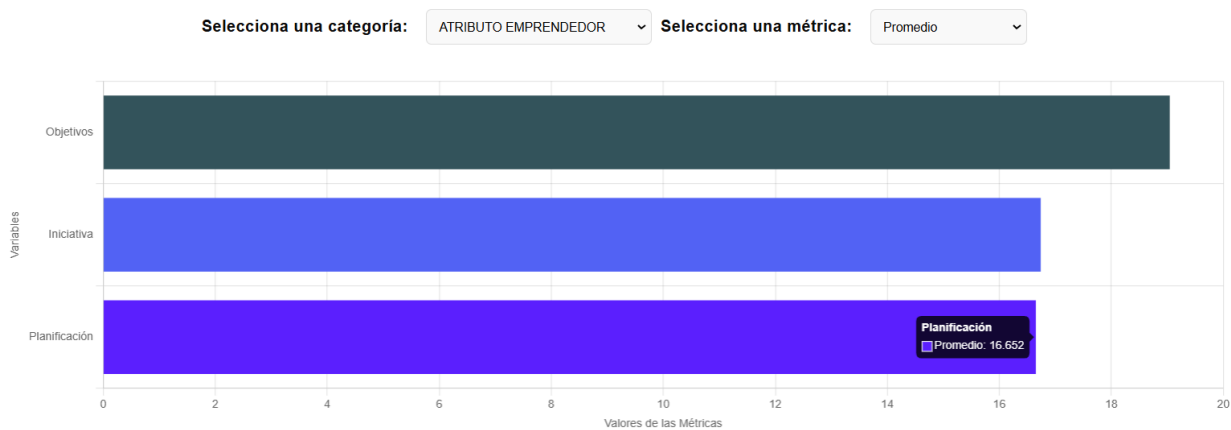


Ilustración 4. 29 Grafico de Comparación de las Estadísticas Grupales (Categorías y Expresiones)

Como se había mencionado antes, para el análisis de la evolución de las competencias emprendedoras a lo largo de la carrera, vamos a hacer la comparación entre los grupos de **primer y octavo semestre** de la carrera de **Ingeniería en Innovación Agrícola Sustentable del ITESCAM**. A través de la plataforma, los docentes pueden acceder a la sección de **“Comparar Estadísticas entre Grupos”**, donde es posible seleccionar dos grupos, el cuestionario de evaluación y la métrica que queremos analizar.



Yuridia Caamal Euan
caamal21@gmail.com

- Home
- Grupos
- Comparacion de grupos
- Cuestionarios aplicados
- Alumnos
- Cerrar Sesión

INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI

Carrera: INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE

Semestre: 1º Grupo GRUPO 1 "A"

Código de Acceso: FYP9GvMc

[Ver Integrantes](#)

✓ CUESTIONARIOS APLICADOS



Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs)²

Fecha de aplicación: 2025-01-28

[Ver Estadísticas](#)

Ilustración 4. 30 Información del Grupo de Primer Semestre (Sujetos de Prueba)



Yuridia Caamal Euan
caamal21@gmail.com

- Home
- Grupos
- Comparacion de grupos
- Cuestionarios aplicados
- Alumnos
- Cerrar Sesión

INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI

Carrera: INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE

Semestre: 8º Grupo GRUPO 8"A"

Código de Acceso: 4wdOBonu

[Ver Integrantes](#)

✓ CUESTIONARIOS APLICADOS



Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs)²

Fecha de aplicación: 2023-05-17

[Ver Estadísticas](#)

Ilustración 4. 31 Información del Grupo de Octavo Semestre (Sujetos de Prueba)

 Yuridia Caamal Euan caamal21@gmail.com Home Grupos Comparacion de grupos Cuestionarios aplicados Alumnos Cerrar Sesión	Administración de Alumnos					
	+ BUSCAR Y AÑADIR ALUMNO		Exportar a PDF			
	NOMBRE	PRIMER APELLIDO	SEGUNDO APELLIDO	CARRERA	INSTITUCION	CORREO
	Andre Jasiel	Caamal	Gomez	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	AndreCaamal7@gr
	Bárbara Isabel	Pool	Cauich	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	PoolBarbara43@gr
	Ana Cristina	Chan	Tamay	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	ChanAna71@gmail
	Santiago Aniel	Pech	Miss	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	PechSantiago58@g
	Abimael	Bacab	Keb	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	BacabAbimael57@c
	Gabriela Esmeralda	Tzuc	Colli	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	TzucGabriela51@gr
	Jair Alexis	Fernández	Tun	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	FernandezJair72@c
	Paola Berenice	Rodríguez	Mata	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	RodriguezPaola86@c
	Seidy Lizeth	Cauich	Chan	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	CauichSeidy94@gr

Ilustración 4. 32 Tabla con la información de los alumnos de prueba

Administración de Alumnos					
+ BUSCAR Y AÑADIR ALUMNO		Exportar a PDF			
NOMBRE	PRIMER APELLIDO	SEGUNDO APELLIDO	CARRERA	INSTITUCION	
Andre Jasiel	Caamal	Gomez	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	
Bárbara Isabel	Pool	Cauich	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	
Ana Cristina	Chan	Tamay	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	
Santiago Aniel	Pech	Miss	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	
Abimael	Bacab	Keb	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	
Gabriela Esmeralda	Tzuc	Colli	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	
Jair Alexis	Fernández	Tun	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	
Paola Berenice	Rodríguez	Mata	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	
Seidy Lizeth	Cauich	Chan	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	

Ilustración 4. 33 Tabla con la información de los alumnos de prueba

Administración de Alumnos

Exportar a PDF

ID	SEGUNDO APELLIDO	CARRERA	INSTITUCION	CORREO	SEMESTRE	ACCIONES
	Gomez	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	AndreCaamal7@gmail.com	8°	 
	Cauich	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	PoolBarbara43@gmail.com	8°	 
	Tamay	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	ChanAna71@gmail.com	8°	 
	Miss	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	PechSantiago58@gmail.com	8°	 
	Keb	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	BacabAbimael57@gmail.com	8°	 
	Colli	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	TzucGabriela51@gmail.com	8°	 
	Tun	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	FernandezJair72@gmail.com	8°	 
	Mata	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	RodriguezPaola66@gmail.com	8°	 
	Chan	INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE	INSTITUTO TECNOLOGICO SUPERIOR DE CALKINI	CauichSeidy94@gmail.com	8°	 

Ilustración 4. 34 Tabla con la información de los alumnos de prueba



Yuridia Caamal Euan
caamal21@gmail.com

- Home
- Grupos
- Comparacion de grupos
- Cuestionarios aplicados
- Alumnos
- Cerrar Sesión

Resultados de Andre Jasiel Caamal Gomez (7069)

Nombre: Andre Jasiel Caamal Gomez
Correo: AndreCaamal7@gmail.com
Semestre: 8°
Carrera: INGENIERIA EN INNOVACIÓN AGRICOLA SUSTENTABLE

Nombre del cuestionario: Cuestionario Estándar de Autoevaluación de las Características Emprendedoras Personales (CEPs)2
Descripción: Este cuestionario revisa en forma resumida las 10 características emprendedoras personales (CEPs), que forman la esencia de la personalidad emprendedora.
Fecha de publicacion:

Resultado de autoevaluación

Variable	Puntaje
Factor de corrección	23
Buscar y aprovechar las oportunidades	12
Ser Persistentes	12
Cumplimiento de compromisos	17
Exigir eficiencia y calidad	12
Correr riesgos moderados	13
Fijar Objetivos	12
Buscar y conseguir información	16
Planificar y hacer seguimiento sistemático	14
Persuasión y creación de redes de apoyo	10
Tener autoconfianza	12

Ilustración 4. 35 Vista de Puntajes Obtenidos por el alumno

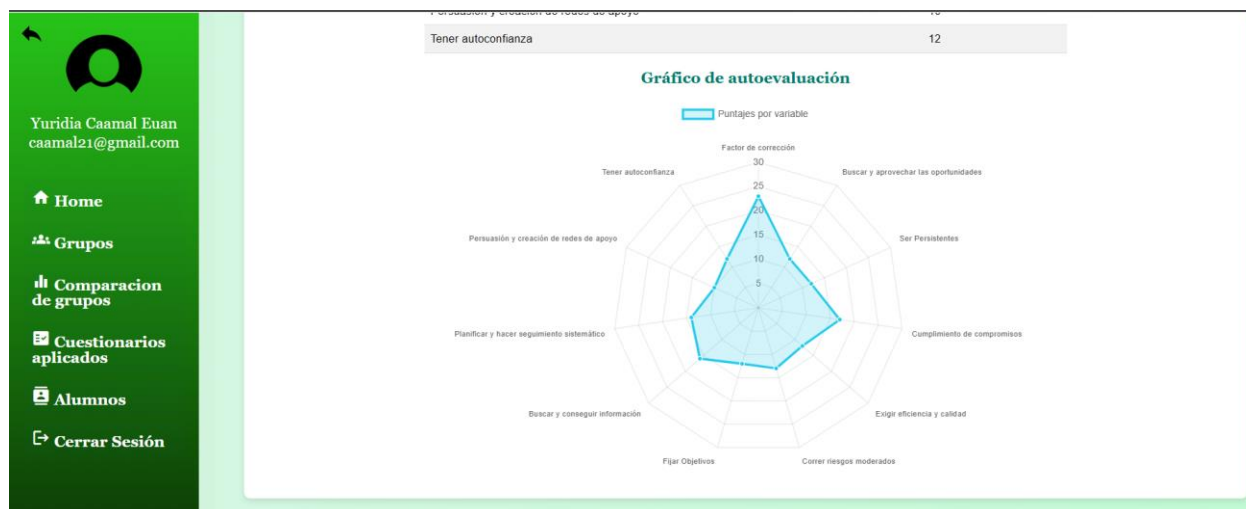


Ilustración 4. 36 Vista de la Grafica araña de los puntajes obtenidos



Ilustración 4. 37 Vista de la Grafica araña de los puntajes obtenidos

La ilustración 4.38 muestra la sección que se utilizara para realizar esta comparación, donde se seleccionaron el grupo de **primer semestre (Grupo 1 “A”)** y el grupo de **octavo semestre (Grupo 8 “A”)**, vamos a seleccionar el cuestionario que se aplicó que en este caso es el **“Cuestionario**

Estándar de Autoevaluación de las Características Emprendedoras” y en este caso utilizaremos la métrica “Promedio”.



Ilustración 4. 38 Resultado de la comparación de los dos grupos de prueba

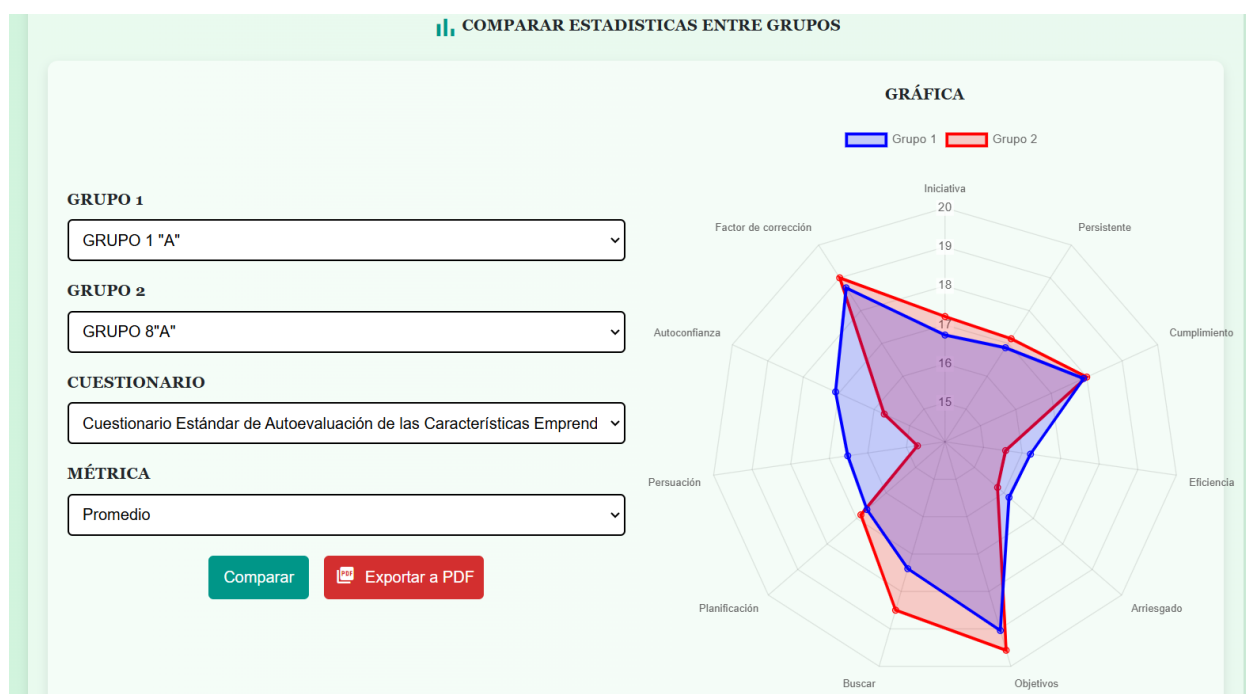


Ilustración 4. 39 Resultado de la comparación de los dos grupos de prueba (Grafica araña)

En esta grafica de araña podemos observar que los estudiantes de octavo semestre presentan puntuaciones más altas en la mayoría de las competencias evaluadas tales como ***“Iniciativa”, “Persistencia”, “Cumplimiento” y “Eficiencia”***. Esto nos sugiere que, a medida que los estudiantes avanzan en la carrera, fortalecen sus competencias emprendedoras, alineándose más con el perfil de egreso. Sin embargo, en algunas dimensiones como ***“Autoconfianza”*** y

“Planificación”, las diferencias entre los grupos no son tan marcados, lo que podría indicar áreas de oportunidad dentro del plan de estudios.

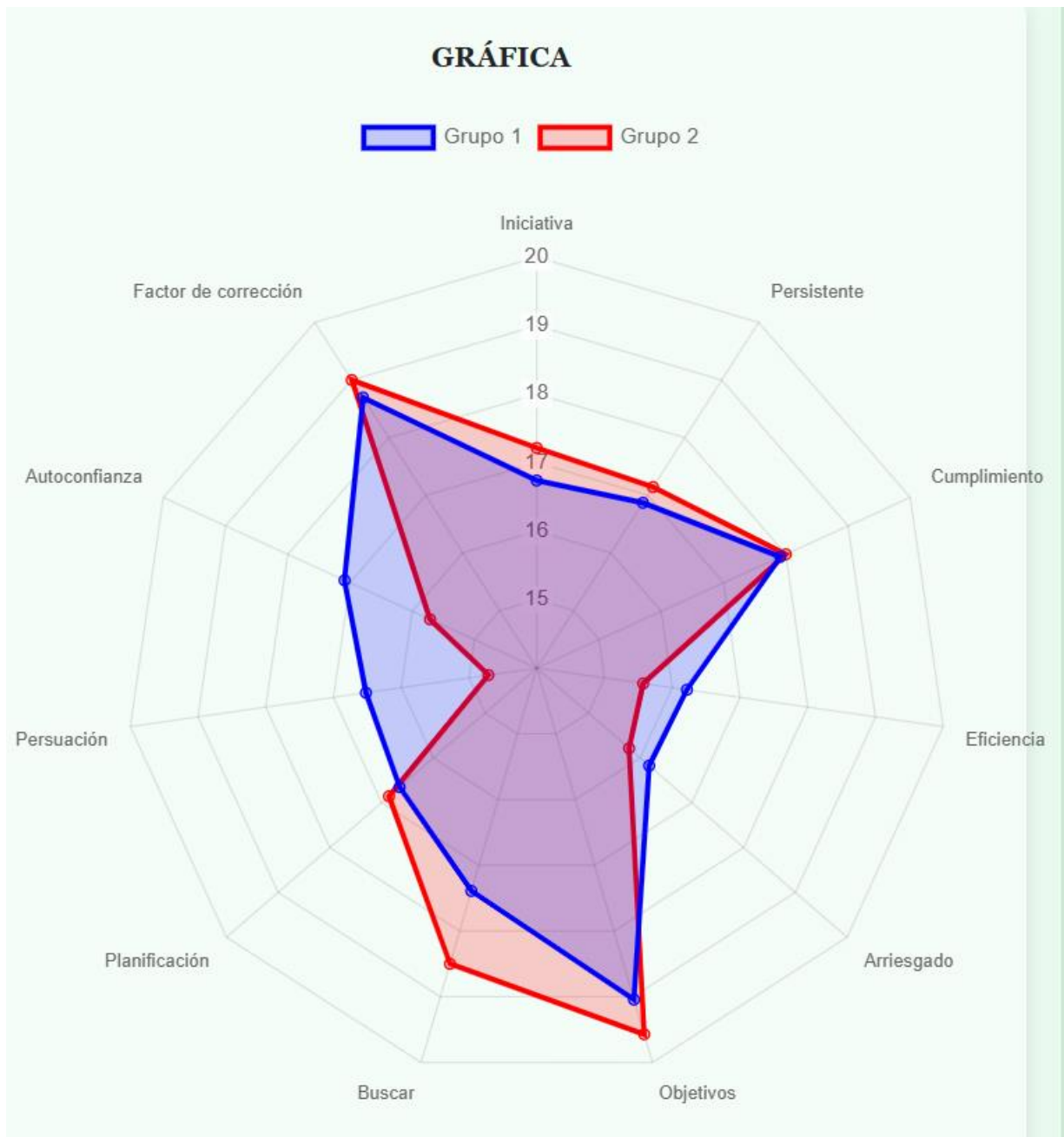


Ilustración 4. 40 Resultado de la comparación de los dos grupos de prueba (Grafica araña)

Para completar nuestro análisis, también se realizó la comparación entre seis estudiantes de primer semestre y seis de octavo semestre. Este análisis nos permite observar la variabilidad individual en el desarrollo de las competencias emprendedoras y detectar los patrones específicos de crecimiento.

En la ilustración 4.41 se muestra la sección que se utilizó para comparar los resultados individuales de los estudiantes, siguiendo el mismo procedimiento que en la comparación grupal.

Ilustración 4. 41 Vista de la información de los alumnos para la comparación

En la primera comparación entre estudiantes podemos observar en la gráfica, que el estudiante de octavo semestre (Alumno 2) muestra un mayor desarrollo en la mayoría de las competencias evaluadas en comparación con el estudiante de primer semestre (Alumno 1). Esto sugiere que, a medida que los estudiantes avanzan en su formación académica, tienden a fortalecer sus capacidades emprendedoras, alineándose progresivamente con el perfil de egreso establecido.

Ilustración 4. 42 Primera comparación entre alumnos (Primer y octavo semestre IIAS)

En particular, se evidencia que el alumno de octavo semestre presenta puntuaciones más altas en dimensiones clave como **“Buscar y aprovechar oportunidades”**, **“Planificar y hacer seguimiento sistemático”**, **“Exigir eficiencia y calidad”** y **“Persuasión y creación de redes de apoyo”**, lo que indica un avance en habilidades estratégicas y de liderazgo. Sin embargo,

algunas competencias, como “Correr riesgos moderados” y “Tener autoconfianza”, no presentan una diferencia significativa, lo que podría sugerir que ciertos aspectos de perfil emprendedor requieren un enfoque más específico dentro del proceso formativo.

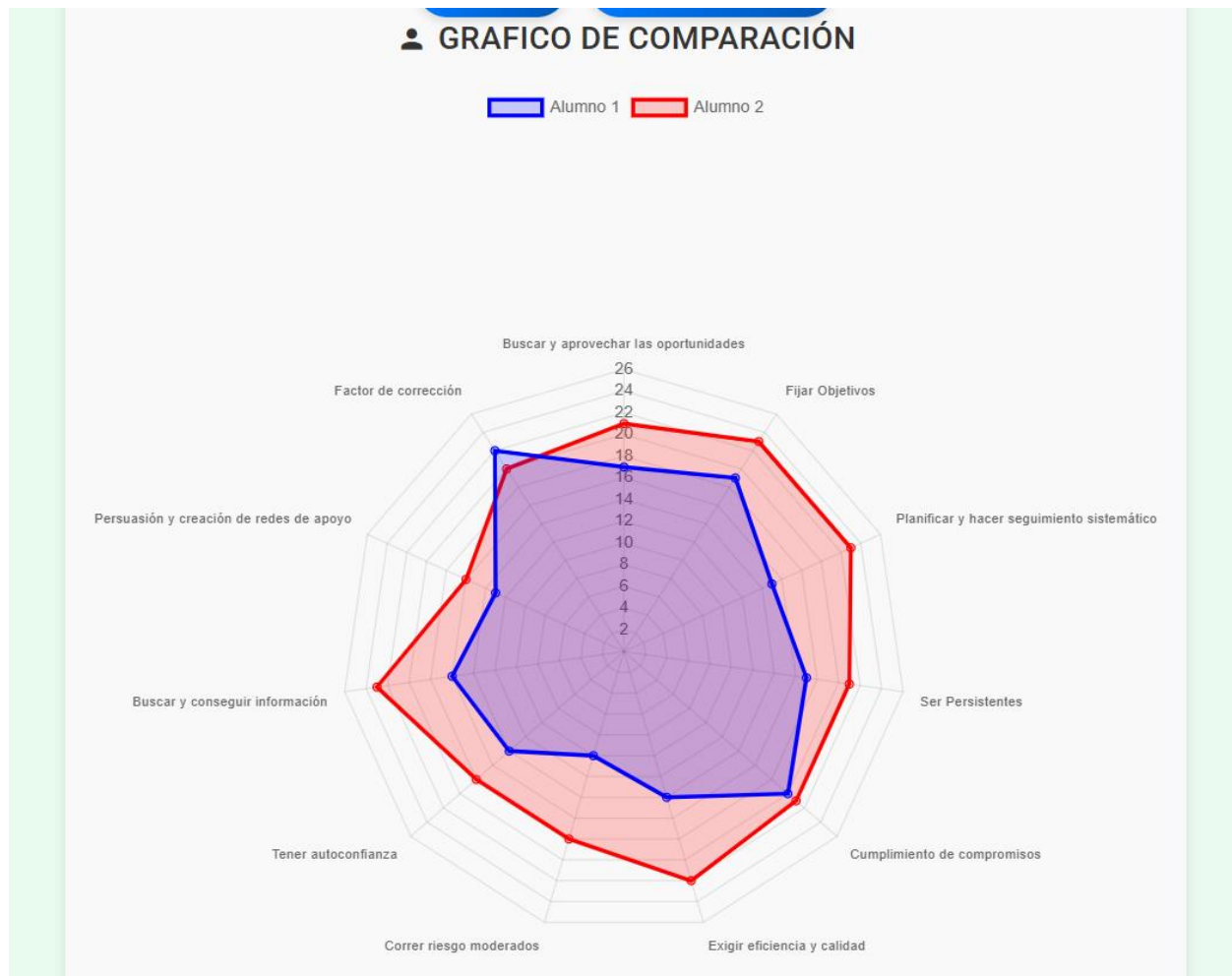


Ilustración 4. 43Primera comparación entre alumnos grafica de araña (Primer y octavo semestre IIAS)

En la segunda comparación entre estudiantes, la gráfica muestra que el Alumno 2 (Octavo semestre) presenta un mayor desarrollo en la mayoría de las competencias evaluadas, lo que confirma la tendencia observada en la primera comparación. Esto nos sugiere que, conforme los estudiantes avanzan en su formación, fortalecen sus habilidades emprendedoras y se alinean mejor con el perfil de egreso de la carrera

Las áreas en las que el Alumno 2 muestra una diferencia notable con respecto al alumno 1 son:

- **Fijar objetivos:** Refleja una mayor claridad en la planificación y establecimiento de metas.
- **Planificar y hacer seguimiento sistemático:** Indica una mejor capacidad para estructurar y monitorear proyectos.
- **Ser persistente:** Sugiere un mayor compromiso y determinación para enfrentar desafíos

- **Cumplimiento de compromisos:** Demuestra una mayor responsabilidad y disciplina en el logro de tareas
- **Exigir eficiencia y calidad:** Indica un enfoque más riguroso hacia la mejora continua y la optimización de recursos.

Estos resultados refuerzan la idea de que la formación académica tiene un impacto positivo en el desarrollo de habilidades estratégicas y organizativas, esenciales para el ejercicio profesional en el sector agrícola. Sin embargo, hay competencias donde la diferencia entre ambos estudiantes es menos marcada o incluso similar:

- **Correr riesgos moderados:** Ambos estudiantes presentan niveles similares, lo que sugiere que la propensión a asumir riesgos puede depender más de la personalidad y experiencias individuales que del proceso formativo.
- **Tener autoconfianza:** Se observa que esta habilidad no se desarrolla de manera significativa con el avance académico, lo que indica la necesidad de estrategias adicionales dentro del programa educativo para fortalecer la seguridad en la toma de decisiones.

COMPARAR ESTADÍSTICAS ENTRE ALUMNOS

Grupo 1	Grupo 2
GRUPO 1 "A" INGENIERIA EN INNOVACIÓN	GRUPO 8"A" INGENIERIA EN INNOVACIÓN
Alumno 1	Alumno 2
Yareni Guadalupe Canul May	Paola Berenice Rodríguez Mata
Cuestionario	
Cuestionario Estándar de Autoevaluación de las Características	
Comparar Exportar a PDF	

Ilustración 4. 44 Segunda comparación entre alumnos (Primer y octavo semestre IIAS)

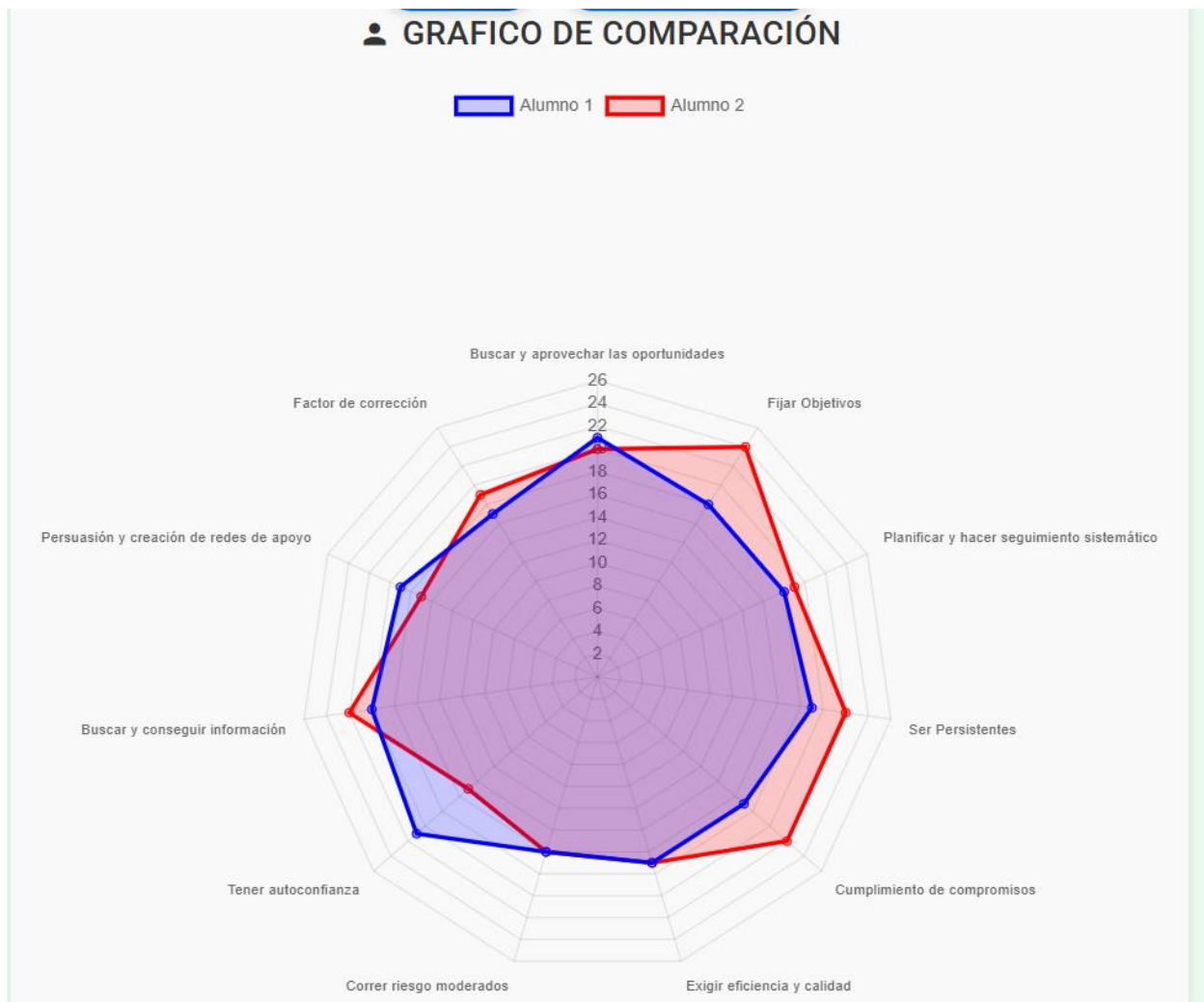


Ilustración 4. 45 Segunda comparación entre alumnos Grafica de araña (Primer y octavo semestre IIAS)

En la tercera comparación entre estudiantes, se observa una tendencia similar a las comparaciones previas, en donde el alumno de octavo semestre (Alumno 2) presenta un mayor desarrollo en la mayoría de las competencias emprendedoras en comparación con el alumno de primer semestre (Alumno 1). En particular, el estudiante de octavo semestre muestra un desempeño superior en dimensiones clave como “Fijar objetivos”, “Planificar y hacer seguimiento sistemático”, “Ser persistentes” y “Exigir eficiencia y calidad”, lo que sugiere un fortalecimiento en las habilidades de planeación y cumplimiento de metas a lo largo de su formación académica.

Asimismo, es notable que el alumno de octavo semestre tiene puntuaciones mas altas en “Cumplimiento de compromisos” y “Correr riesgos moderados”, lo que nos indica una mejor disposición para asumir retos con responsabilidad y tomar decisiones estratégicas en entornos de incertidumbre. Sin embargo, a diferencia de comparaciones previas, en esta ocasión el estudiante de primer semestre muestra un desempeño ligeramente superior en “Persuasión y

creación de redes de apoyo”, lo que podría indicar que ciertas habilidades sociales y de liderazgo pueden desarrollarse de manera independiente a la experiencia académica.

Un aspecto por resaltar es que, al igual que en las comparaciones previas, la diferencia en “Correr riesgos moderados” y “Tener autoconfianza” sigue siendo poco significativa entre ambos estudiantes, lo que sugiere que estas competencias no evolucionan con la misma rapidez que otras habilidades emprendedoras y podrían requerir un enfoque mas específico dentro del programa educativo. Esta comparación refuerza la tendencia de que los estudiantes de octavo semestre presentan un desarrollo más sólido en competencias relacionadas, con la planificación, el seguimiento y la exigencia de calidad, pero también pone en evidencia que ciertas habilidades como la autoconfianza y la toma de riesgos, pueden requerir metodologías distintas para potenciarse en el transcurso de la formación académica.

COMPARAR ESTADÍSTICAS ENTRE ALUMNOS

Grupo 1

GRUPO 1 "A" INGENIERIA EN INNOVACIÓ ▾

Alumno 1

Mayte Jael Naal Ucan ▾

Grupo 2

GRUPO 8"A" INGENIERIA EN INNOVACIÓ ▾

Alumno 2

Santiago Aniel Pech Miss ▾

Cuestionario

Cuestionario Estándar de Autoevaluación de las Características ▾

Comparar **Exportar a PDF**

Ilustración 4. 46 Tercera comparación entre alumnos (Primer y octavo semestre IIAS)

GRAFICO DE COMPARACIÓN

Alumno 1 Alumno 2

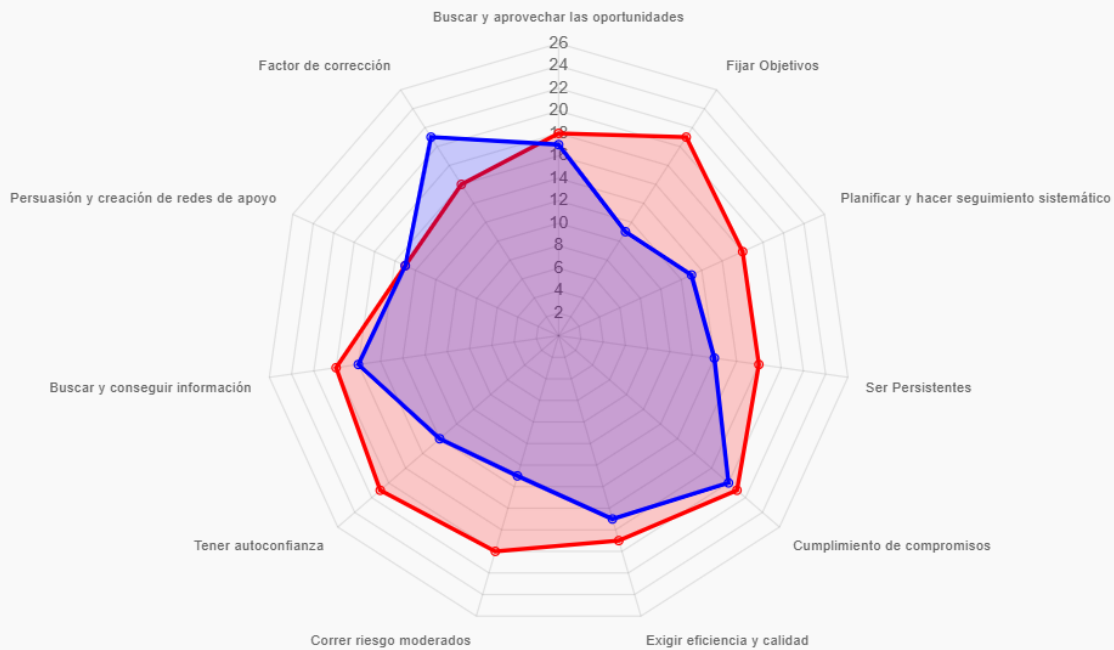


Ilustración 4. 47 Tercera comparación entre alumnos Grafica de araña (Primer y octavo semestre IIAS)

En la cuarta comparación que se realizó, se observa que el alumno de octavo semestre presenta un desarrollo más avanzado en la mayoría de las características emprendedoras en comparación con el alumno de primer semestre, lo que indica una mayor cercanía con el perfil de egreso de la carrera de ingeniería en Innovación Agrícola Sustentable.

En el atributo emprendedor, el estudiante de octavo semestre muestra mejores resultados en “Buscar y aprovechar las oportunidades”, “Fijar objetivos” y “Planificar y hacer seguimiento sistemático”. Esto significa que ha desarrollado una mayor capacidad para identificar oportunidades dentro del sector agrícola, definir metas estratégicas y planificar proyectos a largo plazo, lo cual es fundamental para diseñar y operar empresas agrícolas eficientes.

En la actitud emprendedora, el estudiante de octavo semestre sobresale en “Ser persistente”, “Cumplimiento de compromisos” y “Exigir eficiencia y calidad”, lo que refleja una mayor preparación para enfrentar desafíos, mantener un compromiso con el sector agrícola y buscar la optimización en los sistemas de producción. Esto es clave para su rol como asesor y organizador de productores agrícolas.

En la capacidad emprendedora, se observa que el alumno de octavo semestre tiene una ventaja en “Correr riesgos moderados” y “Tener autoconfianza”. Su disposición a asumir riesgos controlados y su seguridad en la toma de decisiones sugieren que está más preparado para participar en la solución de problemas del sector agrícola, evaluando estrategias de producción y comercialización sin comprometer los recursos naturales.

Sin embargo, en la capacidad interpersonal, el alumno de primer semestre tiene una ventaja en “Persuasión y creación de redes de apoyo” y “Factor de corrección”. Esto indica que, aunque el estudiante de octavo semestre está más alineado con el perfil de egreso en términos de planificación y gestión de proyectos, el de primer semestre podría tener una mayor facilidad para comunicarse, influir en otros y recibir retroalimentación, lo que también es una competencia valiosa dentro del sector agrícola.

En general, esta comparación confirma que el estudiante de octavo semestre está más cercano al logro del perfil de egreso, especialmente en las competencias relacionadas con la planificación estratégica, la eficiencia en la producción y la toma de decisiones con seguridad. Sin embargo, el desarrollo de habilidades interpersonales, como la persuasión y la creación de redes, podría requerir un mayor énfasis en la formación académica para fortalecer aún más su preparación.

COMPARAR ESTADÍSTICAS ENTRE ALUMNOS

Grupo 1

GRUPO 1 "A" INGENIERIA EN INNOVACIÓN

Grupo 2

GRUPO 8"A" INGENIERIA EN INNOVACIÓN

Alumno 1

Jorge Carlos Valle Haas

Alumno 2

Jesus David Suárez Kantun

Cuestionario

Cuestionario Estándar de Autoevaluación de las Características

Comparar Exportar a PDF

Ilustración 4. 48 Cuarta comparación entre alumnos (Primer y octavo semestre IIAS)

GRAFICO DE COMPARACIÓN

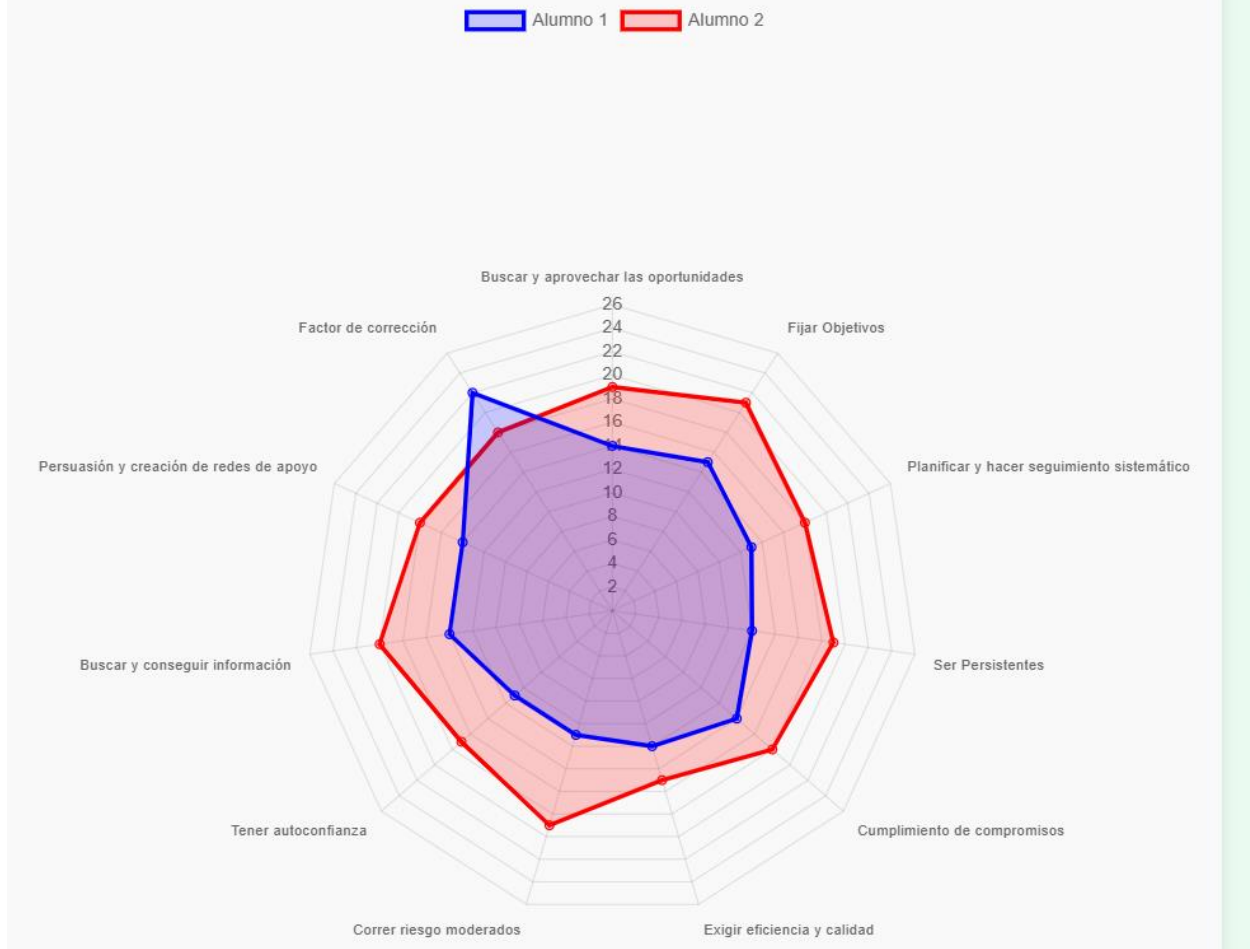


Ilustración 4. 49 Cuarta comparación entre alumnos Gráfica de araña (Primer y octavo semestre IIAS)

En la quinta comparación, los resultados muestran nuevamente una tendencia en la que el estudiante de octavo semestre presenta un mayor desarrollo en la mayoría de las competencias emprendedoras en relación con el perfil de egreso. En el apartado de atributo emprendedor, destaca su superioridad en la capacidad de fijar objetivos y planificar de manera sistemática, lo que indica un mejor enfoque estratégico para la gestión de proyectos agrícolas. En cuanto a la actitud emprendedora, la persistencia y el cumplimiento de compromisos están más desarrollados en el estudiante avanzado, reflejando su mayor responsabilidad en la ejecución de actividades profesionales.

En la capacidad emprendedora, la diferencia en autoconfianza y disposición para asumir riesgos sigue siendo notable, lo que sugiere que el estudiante de octavo semestre ha adquirido una mayor seguridad y madurez en la toma de decisiones, crucial para enfrentar los desafíos del sector agrícola. Sin embargo, en la capacidad interpersonal, aunque el estudiante más avanzado mantiene la ventaja en la búsqueda de información, el de primer semestre muestra un desempeño

competitivo en la persuasión y creación de redes de apoyo, lo que indica que aún en etapas iniciales hay fortalezas en la interacción social y el liderazgo.

COMPARAR ESTADÍSTICAS ENTRE ALUMNOS

Grupo 1

GRUPO 1 "A" INGENIERIA EN INNOVACIÓ

Grupo 2

GRUPO 8"A" INGENIERIA EN INNOVACIÓ

Alumno 1

Eiffy Noeli Kuk Euan

Alumno 2

Daniel David Uc Tun

Cuestionario

Cuestionario Estándar de Autoevaluación de las Característic

Comparar

Exportar a PDF

Ilustración 4. 50 Quinta comparación entre alumnos (Primer y octavo semestre IIAS)

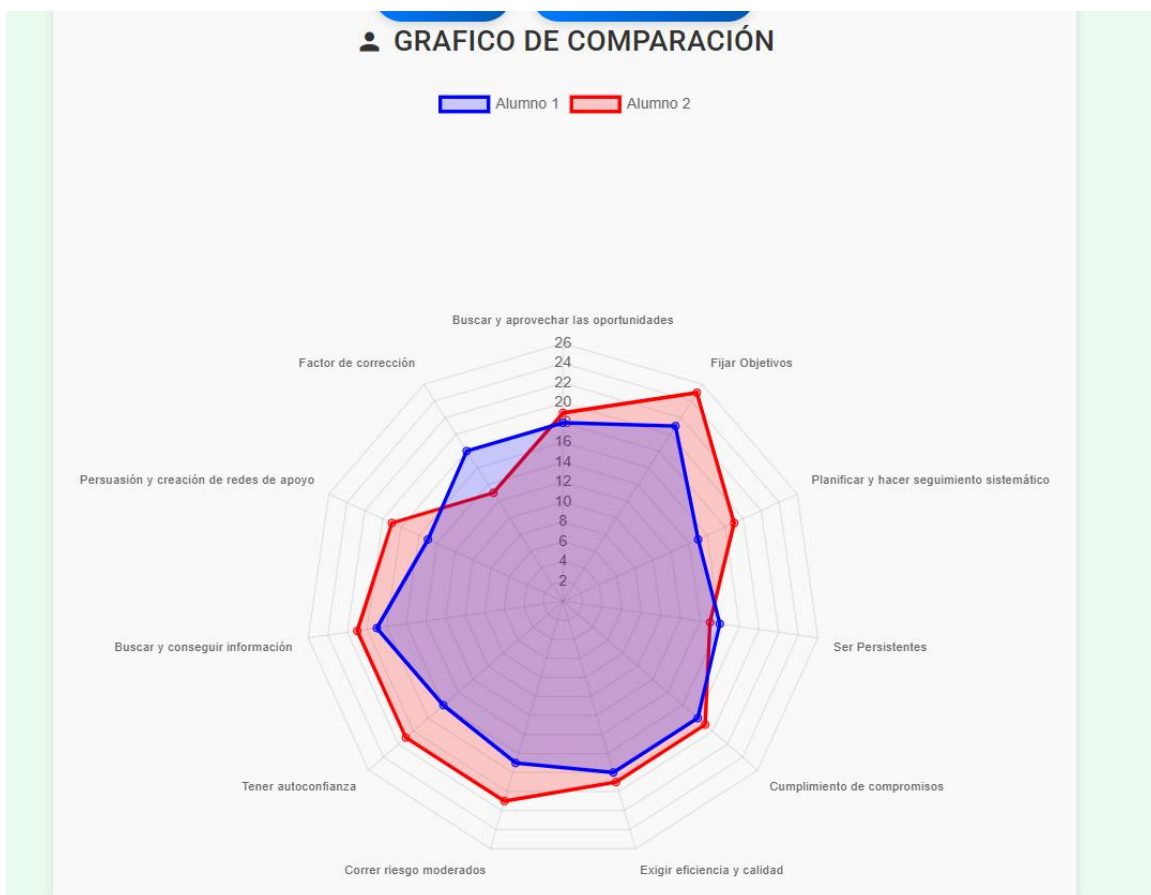


Ilustración 4. 51 Quinta comparación entre alumnos Grafica de araña (Primer y octavo semestre IIAS)

En la sexta y última comparación, se reafirma la tendencia de que el estudiante de octavo semestre ha desarrollado competencias emprendedoras en mayor medida que el de primer semestre, lo que lo acerca más al perfil de egreso de Ingeniería en Innovación Agrícola Sustentable. Se observa un avance significativo en la capacidad de fijar objetivos, planificar y hacer seguimiento sistemático, elementos esenciales dentro del apartado de gestión y liderazgo. Asimismo, en términos de actitud emprendedora, el estudiante de octavo semestre muestra una mayor persistencia y cumplimiento de compromisos, reflejando una madurez profesional alineada con las exigencias del perfil de egreso.

En cuanto a la capacidad de identificar oportunidades y generar redes de apoyo, el estudiante avanzado supera a su contraparte en la búsqueda de información y persuasión, lo que evidencia un mejor aprovechamiento de los recursos y un enfoque más estratégico. Sin embargo, aunque las diferencias siguen siendo marcadas, el estudiante de primer semestre demuestra un desarrollo incipiente en varias áreas, lo que sugiere que, con el adecuado acompañamiento y formación, puede cerrar la brecha en su desarrollo emprendedor. En general, la progresión observada confirma que la formación académica y la experiencia acumulada en la carrera favorecen el cumplimiento de las competencias del perfil de egreso.

COMPARAR ESTADÍSTICAS ENTRE ALUMNOS

Grupo 1

GRUPO 1 "A" INGENIERIA EN INNOVACIÓ ▾

Alumno 1

Rogers Riquelme Herrera May ▾

Grupo 2

GRUPO 8"A" INGENIERIA EN INNOVACIÓ ▾

Alumno 2

Cesar Osvaldo Pech Bonilla ▾

Cuestionario

Cuestionario Estándar de Autoevaluación de las Características ▾

Comparar **Exportar a PDF**

Ilustración 4. 52 Sexta comparación entre alumnos (Primer y octavo semestre IIAS)

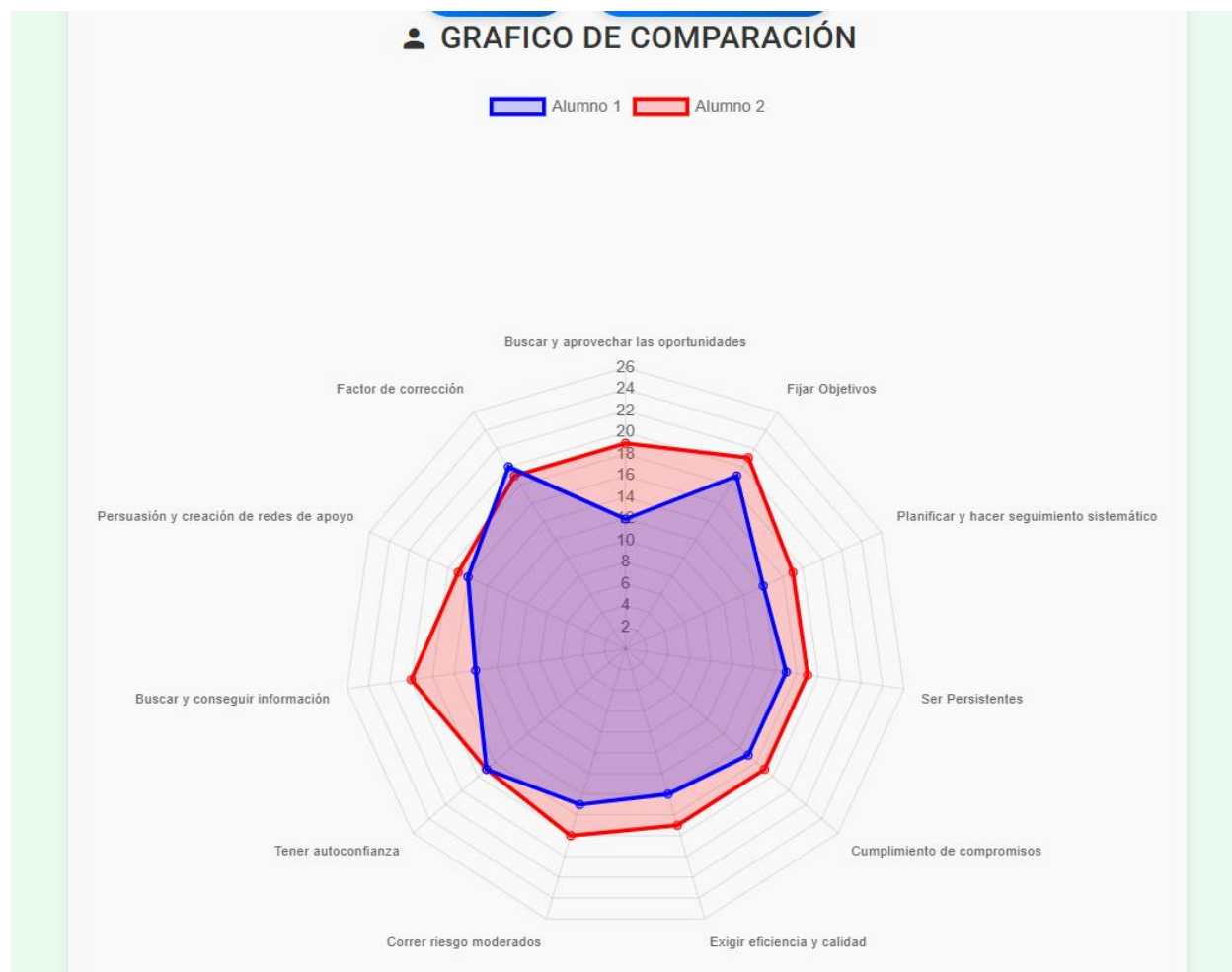


Ilustración 4. 53 Sexta comparación entre alumnos Gráfica de araña (Primer y octavo semestre IIAS)

Al analizar los resultados individuales, se reafirma la tendencia observada en la comparación grupal: los estudiantes de octavo semestre tienden a obtener puntuaciones más altas en la mayoría de las competencias evaluadas. Esta evolución es coherente con el proceso formativo de la carrera de Ingeniería en Innovación Agrícola Sustentable, donde se espera que los estudiantes adquieran y fortalezcan habilidades clave a lo largo de su trayectoria académica.

Las gráficas de comparación reflejan que los alumnos de octavo semestre muestran fortalezas significativas en aspectos como la fijación de objetivos, la planificación y el seguimiento sistemático, lo que demuestra una mayor madurez en la gestión y ejecución de proyectos. Además, se observa un nivel superior de persistencia y cumplimiento de compromisos, características esenciales para el éxito en entornos emprendedores y altamente valoradas dentro del perfil de egreso. En contraste, los estudiantes de primer semestre presentan puntuaciones más bajas en estas áreas, lo que sugiere una menor experiencia en la aplicación práctica de estas competencias.

Otro aspecto relevante es la capacidad de identificar oportunidades y generar redes de apoyo. En esta dimensión, los estudiantes avanzados superan a sus compañeros de primer semestre en la búsqueda de información, persuasión y generación de contactos estratégicos. Esto indica que, con

el paso del tiempo y la acumulación de conocimientos y experiencias, los alumnos desarrollan una visión más amplia y estratégica del entorno en el que pueden desempeñarse profesionalmente. No obstante, algunos estudiantes de primer semestre muestran un desarrollo superior al promedio de su grupo en ciertas habilidades, lo que indica que factores individuales, como experiencias previas o motivación personal, pueden influir en el desarrollo de competencias emprendedoras. Este análisis detallado nos permitió identificar las fortalezas y áreas de mejora tanto a nivel grupal como individual, proporcionando información clave para la implementación de estrategias formativas que potencien el desarrollo emprendedor de los estudiantes.

Capítulo 5. CONCLUSIONES Y RECOMENDACIONES

5.1 Conclusiones

*El desarrollo del sistema de inteligencia de negocios para la identificación y análisis de competencias emprendedoras en estudiantes universitarios permitió evaluar de manera efectiva la evolución de dichas competencias a lo largo de la formación académica. A través del análisis comparativo de datos entre estudiantes de primer y octavo semestre de la carrera de **Innovación Agrícola Sustentable del ITESCAM**, se logró identificar patrones clave en la manera en que los alumnos perciben y desarrollan sus habilidades emprendedoras en función de su perfil de egreso.*

Los resultados del análisis revelaron que, si bien los estudiantes de semestres avanzados deberían estar más alineados con el perfil de egreso de su carrera, en algunos casos, sus puntajes en ciertas competencias disminuyen con el tiempo. Este hallazgo sugiere que, conforme los estudiantes adquieren más experiencia académica y profesional, su percepción sobre habilidades emprendedoras se ajusta de una visión aspiracional a una más estratégica y realista.

*En particular, se identificó que competencias como **“Buscar y aprovechar oportunidades”** y **“Persuasión y creación de redes de apoyo”** presentaron puntajes más altos en algunos estudiantes de primer semestre en comparación con los de octavo semestre, lo que indica la necesidad de un enfoque más integral en la formación emprendedora, asegurando que los estudiantes no solo desarrollen habilidades técnicas, sino que también mantengan la confianza y motivación hacia el emprendimiento.*

Asimismo, la plataforma desarrollada ha demostrado su utilidad al proporcionar una herramienta eficiente para la autoevaluación y análisis de competencias emprendedoras, facilitando el acceso a reportes detallados y visualizaciones dinámicas de los datos. La inclusión de dashboards personalizados para alumnos, docentes y administradores optimiza la experiencia del usuario, asegurando que la información obtenida sea clara y accesible.

En términos de validación de la hipótesis, los resultados confirmaron que el sistema de inteligencia de negocios puede identificar patrones en la evolución de las competencias emprendedoras de los estudiantes universitarios en un estudio longitudinal, basándose en su perfil de egreso. Sin embargo, también evidenciaron que el desarrollo de estas competencias no sigue un patrón uniforme, lo que sugiere la necesidad de fortalecer estrategias de formación y acompañamiento a lo largo de la carrera.

Además, durante el desarrollo del sistema se llevaron a cabo pruebas de usabilidad y exploratorias lo que permitió optimizar su funcionalidad y garantizar una interfaz intuitiva para los diferentes tipos de usuarios. Estas pruebas también ayudaron a identificar áreas de mejora en la presentación de los reportes y la accesibilidad de la información.

En conclusión, este estudio demuestra que la integración de herramientas tecnológicas y análisis de datos en la educación superior puede proporcionar información valiosa para la toma de decisiones académicas y el diseño de estrategias de mejora. El sistema que se ha desarrollado no

solo facilita la identificación de patrones en las competencias emprendedoras, sino que también ofrece una base sólida para la implementación de planes de acción que contribuyan a fortalecer la preparación emprendedora de los futuros egresados.

5.2 Limitaciones y propuestas de mejora

Si bien las limitaciones planteadas inicialmente fueron atendidas, algunas áreas representan oportunidades para futuras mejoras:

- *La plataforma actualmente funciona únicamente como un sitio web. Una posible evolución del sistema sería convertirlo en una aplicación móvil, ampliando su accesibilidad.*
- *Aunque los usuarios deben ser registrados con matrícula proporcionada por las instituciones, se podría explorar la incorporación de mecanismos de autogestión.*
- *El requisito de acceso a internet puede ser mejorado implementando opciones offline sincronizables.*

Recomendaciones

- ❖ *Capacitar a los usuarios finales para asegurar un correcto registro y uso del sistema.*
- ❖ *Evaluar continuamente la funcionalidad y experiencia de usuario mediante retroalimentación de las instituciones participantes.*
- ❖ *Expandir el alcance del sistema para incluir otros niveles educativos o sectores más allá del ámbito universitario.*
- ❖ *Fortalecer estrategias de formación complementaria, basadas en los hallazgos del sistema, para así asegurar que los estudiantes no solo desarrollen habilidades técnicas, sino que también mantengan la confianza y motivación hacia el emprendimiento.*

Finalmente, el desarrollo de este proyecto no solo cumplió con los objetivos técnicos, sino que también aportó significativamente a nuestra formación profesional, consolidando nuestras habilidades en desarrollo de software, inteligencia de negocios y metodologías ágiles. Además, establece un precedente para el uso de herramientas de análisis de datos en la educación superior, con el potencial de contribuir al diseño de programas académicos más alineados con las necesidades y expectativas del mercado laboral.

Referencias Bibliográficas.

- (COPAES), C. p. (2019). *Normas de Acreditación de Programas Académicos*. Mexico: COPAES.
- (GEM), G. E. (2021). *Emprendimiento en México: Tendencias y Desafíos*. Mexico: GEM México.
- (SEP), S. d. (2022). *Modelo Educativo 2030*. Mexico: SEP.
- ¿Quees? (20 de Marzo de 2023). *La investigación longitudinal es uno de los tipos de estudios más empleados*. Obtenido de ¿Qué es?: <https://quees.mobi/investigacion/investigacion-longitudinal/>
- ANUIES. (2021). *Seguimiento de egresados en México: Análisis de trayectorias laborales*. Mexico.
- Bernal, R. (09 de enero de 2024). *3 consejos para ser un emprendedor persuasivo*. Obtenido de Coach para Emprendedores: <https://coachparaemprendedores.com/emprendedor-persuasivo/#:~:text=La%20persuasi%C3%B3n%20y%20las%20redes%20de%20apoyo%20est%C3%A1n%20relacionadas%20con,hacerle%20frente%20a%20los%20desaf%C3%ADos.>
- Blanco, R. (20 de Mayo de 2021). *Persistencia de un emprendedor*. Obtenido de Emprendimienito: <https://www.emprendimiento.com/post/persistencia-emprendedor>
- Blog, A. (29 de Marzo de 2023). *El perfil de las personas emprendedoras mexicanas: ¿cómo es y qué buscan?* - ASEM - Asociación de Emprendedores de México. Obtenido de ASEM - Asociación de Emprendedores de México: https://asem.mx/blog_asem/el-perfil-del-emprendedor-mexicano-como-es-y-que-busca/?utm_source=google&utm_medium=cpc&utm_campaign=20054955505&utm_term=154193812728&gad_source=1&gclid=CjwKCAiAqrG9BhAVEiwAaPu5zkOfJEQ7hZm3DxuLHOID4sfk9xntKpPzsgahChY_K6UkIYMys2
- Bloom, E. (7 de julio de 2024). *Escala de Likert: que es y ejemplos de como utilizarla*. Obtenido de Blog de E-learning: <https://www.ispring.es/blog/escala-de-likert>
- Camargo Vega, J. J., Joyanes Aguilar, L., & Giraldo Marín, L. M. (1-4 de 2016). La inteligencia de negocios como una herramienta en la gestión académica. *Revista Científica*.
- (2013). *Características emprendedoras personales (CEP)*. Revista Ciencia UNEMI.
- Carbajal, A. (19 de 10 de 2023). *México Emprende*. Obtenido de Emprender: La Fuerza Transformadora del Empresario Mexicano: <https://www.mexicoemprende.org.mx/que-es-emprender/>
- Carbajal, A. (26 de 10 de 2026). *México Emprende*. Obtenido de Inteligencia de Negocios: Un Enfoque Imprescindible para el Éxito Empresarial: <https://www.mexicoemprende.org.mx/inteligencia-de-negocios-un-enfoque-imprescindible-para-el-exito-empresarial/>
- Castillo, I. (31 de Agosto de 2022). *Investigación longitudinal*. Obtenido de Lifeder: <https://www.lifeder.com/investigacion-longitudinal/>

- Cavero, J. (11 de agosto de 2023). *Búsqueda de información. 3ª Competencia emprendedora*. Obtenido de mentorDay WikiTips: <https://mentorday.es/wikitips/busqueda-de-informacion/#:~:text=La%20b%C3%BAqueda%20de%20informaci%C3%B3n%20es,sean%20exitosos%20en%20el%20mercado>.
- Cavero, J. (11 de agosto de 2023). *Cumplimiento de compromisos*. Obtenido de mentorDay WikiTips: <https://mentorday.es/wikitips/cumplimiento-de-compromisos/>
- Chisparural*. (02 de 03 de 2022). Obtenido de Las características de toda persona emprendedora: <https://www.chisparural.gt/articulos/las-caracteristicas-de-toda-persona-emprendedora#:~:text=Las%20personas%20empendedoras%20tienen%20claros,tener%20objetivos%20nos%20vuelve%20inm%C3%B3viles>.
- Claudia García Ancira, A. T. (2020). *Las competencias universitarias y el perfil de egreso*. Obtenido de SciELO Revista Estudios del Desarrollo Social: Cuba y América Latina: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S2308-01322020000100003
- coneval*. (s.f.). Obtenido de Incubadora de Evaluación de Impacto: https://www.coneval.org.mx/EvaluacionDS/PP/Políticas/Impacto/Paginas/incubadora_impacto.aspx
- del Pino Acuña, F. A., Fritzchmann Sánchez, F. D., & Espindola Ortega, A. E. (Julio-Agosto de 2023). *Emprendimiento en el Contexto de la Educación Superior: una Revisión de la Literatura*. *Ciencia Latina Revista Científica Multidisciplinar*.
- Eduardo Ahumada Tello, J. M. (3 de 11 de 2015). *Contaduría y Administración. Inteligencia de negocios: estrategia para el desarrollo de competitividad en empresas de base tecnológica*, págs. 127,158.
- Empleo, S. N. (s.f.). *Cualidades que necesitas para ser un buen emprendedor | OLA*. Obtenido de gob.mx: https://www.observatoriolaboral.gob.mx/static/preparate-empleo/Emprendedores_2.html?utm_source
- Gloria Y. Rojas, V. P. (2018). Instrumento para Identificar Características Personales y Didáctica Utilizadas por los Docentes en la Formación de Emprendedores. *Formación universitaria*.
- Gutiérrez Montoya, G. (12 de 2011). Educación emprendedora en la universidad: Educando para el futuro. *RETOS. Revista de Ciencias de la Administración y Economía*.
- Ibero MTY*. (s.f.). Obtenido de 3 Empresas que han implementado Scrum con éxito: <https://iberomty.mx/blog/3-empresas-implementado-scrum-exito/>
- Indeed, E. e. (24 de Junio de 2024). *Características de un emprendedor exitoso*. Obtenido de Indeed: https://mx.indeed.com/orientacion-profesional/desarrollo-profesional/caracteristicas-emprendedor-exitoso?aceid=&gad_source=1&gclid=CjwKCAiAqrG9BhAVEiwAaPu5ztV4KOeqpbZ1jtZzBcCWfOhIjmRJNsgq0NhCVe_ddbium_hyjpq03RoCJcsQAvD_BwE
- INEGI. (2022). *Encuesta Nacional de Ocupación y Empleo (ENOE)*. Mexico: Instituto Nacional de Estadística y Geografía.

- J, A. J. (24 de marzo de 2009). *Emprender con tecnologías de informacion*. Obtenido de Características Emprendedoras Personales (CEPs): <https://adolfoaraujo.com/2009/03/24/24-caracteristicas-emprendedoras-personales-ceps/>
- J., A. J. (8 de Junio de 2018). 2.4.1- *Características emprendedoras personales (CEPS)*. Obtenido de Emprender con Tecnologías de Información : <https://adolfoaraujo.com/2009/03/24/24-caracteristicas-emprendedoras-personales-ceps/>
- Javier Suárez-Álvarez, I. P. (2016). EVALUACIÓN DE LA PERSONALIDAD EMPRENDEDORA: SITUACIÓN ACTUAL y LÍNEAS DE FUTURO. *Papeles del Psicólogo*, págs. 62-68.
- I, L. B. (2021). IMPLEMENTACIÓN Y PRÁCTICA DE SCRUM EN LA ASIGNATURA DE FORMULACIÓN Y EVALUACIÓN DE PROYECTOS EN LA FACULTAD DE CIENCIAS ECONÓMICAS Y ADMINISTRATIVAS DE LA UNIVERSIDAD EL. *Artículos de investigación científica y tecnológica*.
- Laat, I. A. (3 de 1 de 2023). Guía para la implementación de un Programa sistémico Emprendedor para el Instituto Tecnológico de Calkiní, Campeche. *Emprendedurismo y Economía ambiental*.
- Laura Iturbide Galindo, R. R. (01 de Diciembre de 2020). *Las características del emprendedor mexicano: estudio de caso de Lean Startups México*. Obtenido de The Anáhuac Journal: https://www.scielo.org.mx/scielo.php?pid=S2683-26902020000200012&script=sci_arttext&utm
- Lizbeth Alicia González Tamayo, C. D. (2019). *Emprendimiento en el Estado de México. Global Entrepreneurship Monitor 2019*.
- M.A. Alberto Bucardo Castro, M. L. (10 de 04 de 2015). *Hacia una comprensión de los conceptos de emprendedores y empresarios*. Obtenido de <https://www.redalyc.org/journal/6099/609964673011/html/>
- Mariana. (07 de 04 de 2024). *Autoevaluación como Emprendedor: Una Herramienta para el Éxito*. Obtenido de Ciberlinea: <https://www.scribbr.es/citar/generador/folders/63ApnKgncYzAKWe1t8qrn/lists/WoipGPG7rCgZmesScUX8a/fuentes/6qxVNmdEQBzCgMT986S9Hr/editar/>
- Matas, A. (s.f.). *Diseño del formato de escalas tipo Likert: un estado de la cuestión*. Obtenido de SciELO Revista Electrónica de Investigación Educativa: https://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S1607-40412018000100038
- Mugira, A. (21 de Noviembre de 2024). *¿Qué es la escala de Likert y cómo utilizarla?* Obtenido de QuestionPro: <https://www.questionpro.com/blog/es/que-es-la-escala-de-likert-y-como-utilizarla/>
- Omar Alejandro Pérez Cruz, R. P. (2020). *Determinantes de la inserción laboral en egresados universitarios en México*. Obtenido de SciELO RIDE. Revista Iberoamericana para la Investigación y el Desarrollo Educativo: https://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S2007-74672020000200127
- Oré, B. R. (09 de 02 de 2023). Emprendimiento digital en estudiantes de educación superior. *Horizontes Revista de Investigación en Ciencias de la Educación*.

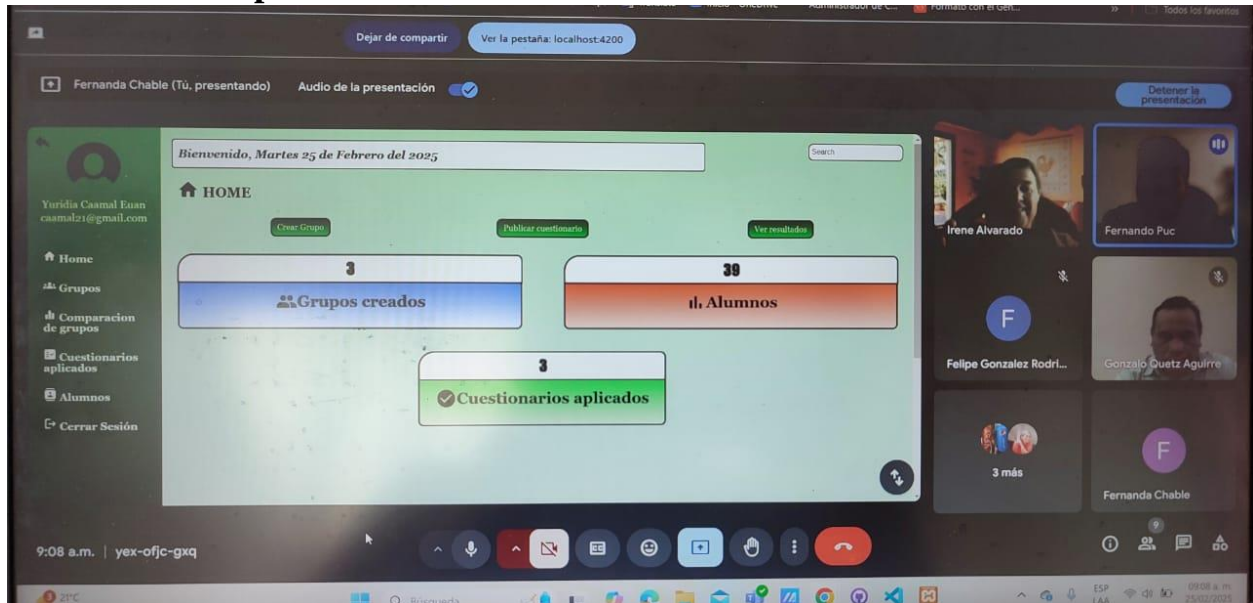
- Pedraza, F. J. (18 de Septiembre de 2023). *Emprende y aprende | Características del emprendedor mexicano*. Obtenido de El sol de tampico OEM:
https://oem.com.mx/elsoldetampico/analisis/emprende-y-aprende-caracteristicas-del-emprendedor-mexicano-primera-parte-16341513?utm_source
- PhD, I. A. (3 de 1 de 2023). Guía para la implementación de un Programa sistémico Emprendedor para el Instituto Tecnológico de Calkiní, Campeche. *Emprendedurismo y Economía ambiental*.
- Poyatos, V. (05 de 11 de 2024). *Aglaia*. Obtenido de Comparación de Power BI con QlikView y Tableau:
<https://aglaia.es/blog/power-bi/comparacion-de-power-bi-con-qlikview-y-tableau/>
- Pública, S. D. (21 de 04 de 2015). *gob.mx*. Obtenido de Modelo de Emprendedores para la Educación Media Superior: <https://www.gob.mx/sep/acciones-y-programas/modelo-de-emprendedores-para-la-educacion-media-superior>
- Que es el emprendimiento*. (27 de 10 de 2023). Obtenido de Aprende Emprende:
<https://aprendeyemprende.mx/que-es-el-emprendimiento>
- Rafael Eduardo Saavedra Leyva, G. M. (2022). Resiliencia del emprendimiento en México. Los casos de las crisis económicas del COVID-19 y subprime. *redalyc.org*, págs. 507-518. Obtenido de Resiliencia del emprendimiento en México. Los casos de las crisis económicas del COVID-19 y subprime: <https://www.redalyc.org/journal/212/21273974008/html/>
- Ramos-Peña, L. G. (2018). *Las Competencias de egreso y su relación con el mercado laboral en el área de las Ciencias Políticas*. Obtenido de Revista política globalidad y ciudadanía:
<https://www.redalyc.org/journal/6558/655869488006/html/>
- Reporte sobre la Complejidad Económica del Estado de Campeche. (julio de 2017). *Estudios de diagnóstico Campeche*, pág. 48.
- Schwaber, K. &. (2020). *Guía de Scrum: La Guía Definitiva de Scrum, el Marco de Trabajo Ágil para Equipos de Proyectos Complejos*.
- Segura, F. J. (13 de mayo de 2019). El perfil emprendedor y la intensidad competitiva del mercado como predictores de supervivencia en microempresas mexicanas. *Contaduría y Administración*, pág. 177.
- Sybing, R. (11 de Febrero de 2025). *Estudios longitudinales: Métodos, ventajas y retos*. Obtenido de ATLAS.ti: <https://atlasti.com/es/research-hub/estudio-longitudinal>
- Tableau. (2024). Obtenido de Qué es el Business Intelligence? Tu guía para la inteligencia de negocios y por qué es importante: <https://www.tableau.com/es-mx/learn/articles/business-intelligence>
- Team, D. E. (20 de Febrero de 2023). *Longitudinal Study: Definition, Pros, and Cons*. Obtenido de Dovetail: <https://dovetail.com/research/longitudinal-study/>
- URUGUAY, U. d. (29 de Octubre de 2021). *Características de los perfiles de egreso – Facultad de Ciencias Sociales*. Obtenido de Ciencias Sociales Universidad de la Republica URUGUAY:
<https://cienciassociales.edu.uy/noticias-ei/caracteristicas-de-los-perfiles-de-egreso/>

Ventana / EBC. (04 de 11 de 2024). Obtenido de El emprendimiento: ¿qué es y para qué sirve? Descubre la clave para que tu proyecto se mantenga a flote:
<https://www.ebc.mx/ventana/emprendimiento-que-es-y-para-que-sirve/>

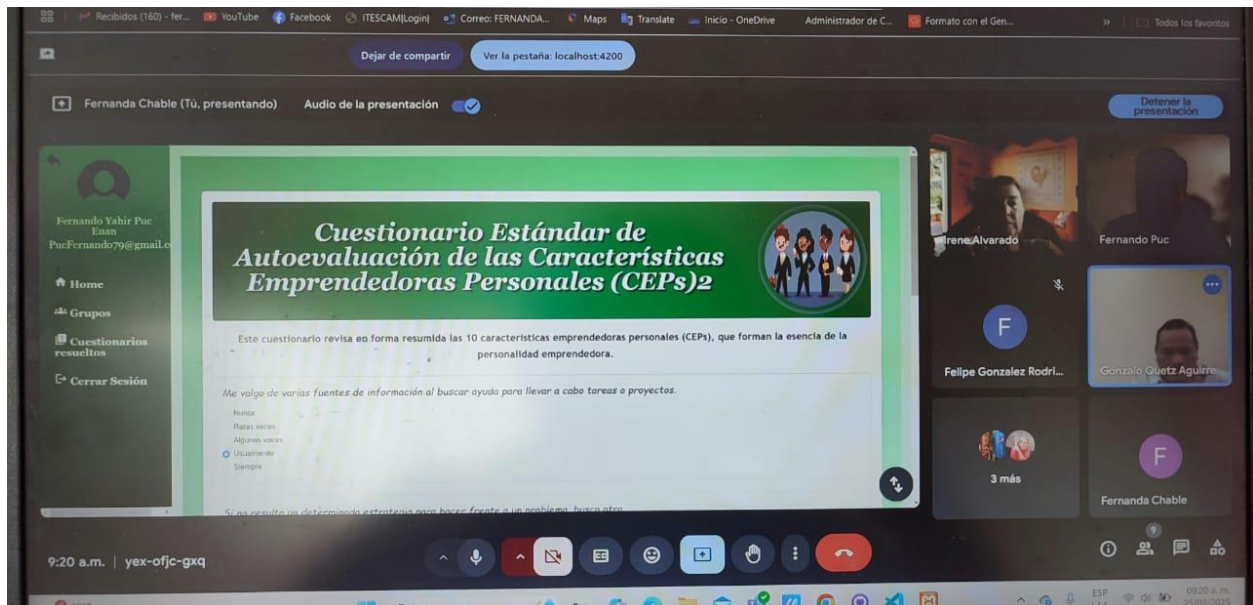
Violeta Martínez Ramírez, A. G. (01 de 01 de 2021). EL IMPACTO DE LA METODOLOGÍA ÁGIL SCRUM IMPLEMENTADO EN EL DESARROLLO DE SOFTWARE EMPRESARIAL.
revistas.milpaalta.tecnm.mx.

ANEXOS.

Pruebas de Aceptación

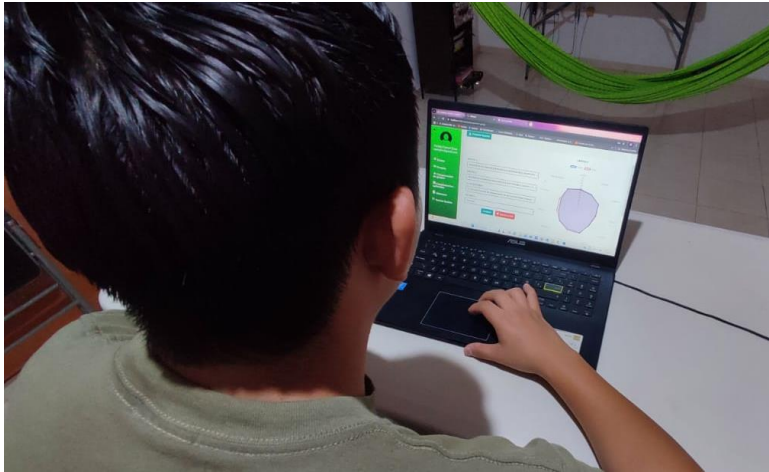


Anexo 1. Reunión de entrega del sistema

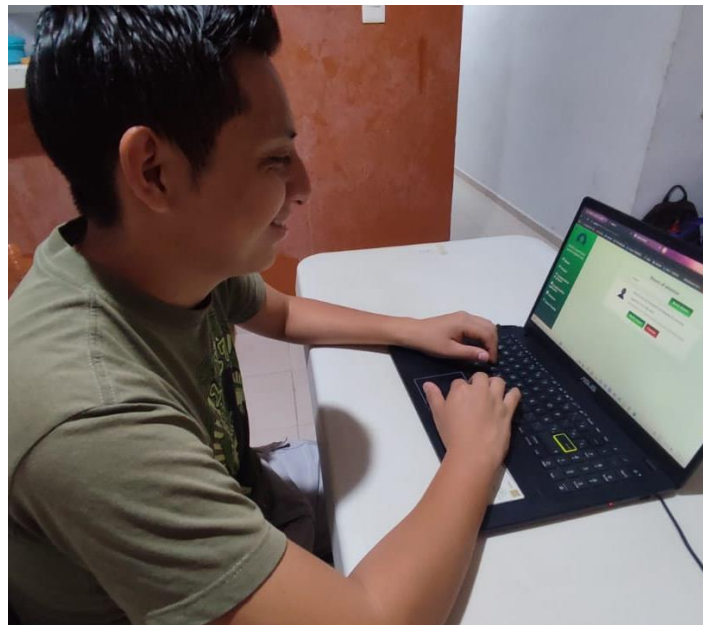


Anexo 2 Reunión de entrega del sistema con el equipo

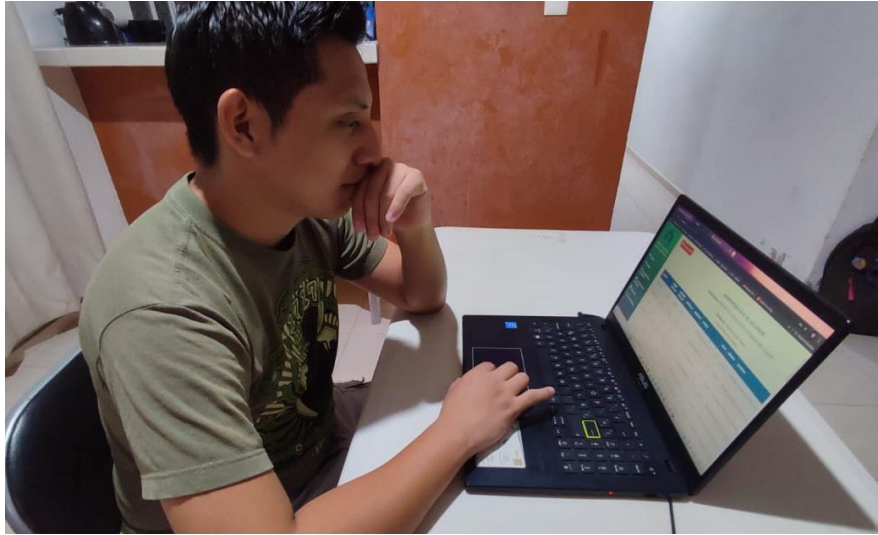
Pruebas de Usabilidad



Anexo 3 Evidencias de las pruebas de usabilidad al sujeto de prueba



Anexo 4 Evidencias de las pruebas de usabilidad al sujeto de prueba



Anexo 5 Evidencias de las pruebas de usabilidad al sujeto de prueba



Anexo 6 Evidencias de las pruebas de usabilidad al sujeto de prueba

Backend

El backend de este proyecto ha sido desarrollado utilizando Laravel como framework principal y se conecta con una base de datos MySQL. Este backend está diseñado para gestionar usuarios, cuestionarios, instituciones, carreras, generaciones, países, grupos, categorías, variables, expresiones y análisis de resultados. A continuación, se describe cada componente con capturas de pantalla representativas del código y las funcionalidades implementadas.

Migraciones

En este proyecto, las migraciones se han utilizado para establecer tablas fundamentales como usuarios, grupos, instituciones, carreras, países, categorías, variables, expresiones, cuestionarios, y sus relaciones. Cada migración define las columnas, tipos de datos, claves primarias, foráneas y restricciones necesarias para garantizar la integridad de los datos. Además, al trabajar con herramientas como Eloquent ORM, las migraciones permiten integrar de manera sencilla el esquema de la base de datos con la lógica de la aplicación, asegurando un flujo de desarrollo eficiente y organizado. A continuación, se describe cada migración creada, junto con su propósito y estructura.

Tablas países

Esta migración crea la tabla *países*, utilizada para almacenar información sobre diferentes países. Aquí definimos un único **IDPaís** como clave primaria, una columna **Nombre** para el nombre del país, y las columnas estándar **created_at** y **updated_at** para el control de tiempos. Su diseño simple nos permitió gestionar datos relacionados con países de manera eficiente y escalable.

```
<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;
class CreatePaisesTable extends Migration
{ public function up()
    { Schema::create('países', function (Blueprint $table) {
        $table->id('IDPaís');
        $table->string('Nombre');
        $table->timestamps();}); }
    public function down()
    {Schema:dropIfExists('países') ;} }
```

Anexo 7. Migración tabla países

Tabla instituciones

Hemos implementado la tabla *instituciones*, diseñada para almacenar información de instituciones educativas. Esta tabla relaciona cada institución con un país mediante la clave foránea **IDPaís**, que se enlaza con la tabla *países*. Además, incluye una columna **Nombre** para identificar cada institución y utiliza restricciones referenciales para mantener la integridad de los datos, como la eliminación en cascada. Esta estructura facilita la gestión de instituciones dentro del sistema.

```

<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateInstitucionesTable extends Migration
{
    public function up()
    {
        Schema::create('instituciones', function (Blueprint $table) {
            $table->id('InstitucionID');
            $table->string('Nombre');
            $table->unsignedBigInteger('IDPais');
            $table->timestamps();
            $table->foreign('IDPais')->references('IDPais')->on('países')
                ->onDelete('cascade'); });
    }

    public function down()
    {
        Schema::dropIfExists('instituciones');
    }
}

```

Anexo 8. Migración de la tabla Instituciones

Tabla carreras

*Hemos creado la tabla carreras, que gestiona la información sobre las diferentes carreras ofrecidas por instituciones educativas. Esta tabla incluye una columna **Nombre** y establece una relación con la tabla instituciones mediante la clave foránea **InstitucionID**, lo que asegura integridad referencial.*


```

<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateCarrerasTable extends Migration
{
    public function up()
    {
        Schema::create('carreras', function (Blueprint $table) {
            $table->id('CarreraID');
            $table->string('Nombre');
            $table->foreignId('InstitucionID') ->constrained('instituciones',
            'InstitucionID') ->onDelete('cascade');
            $table->timestamps(); });
    }

    public function down()
    {
        Schema::dropIfExists('carreras');
    }
}

```

Anexo 9. Migración de la tabla carreras

Tabla generaciones

*La migración para la tabla generaciones estructura la información relacionada con las generaciones académicas de una carrera específica. Cada generación se identifica por su **Nombre** y está vinculada mediante una clave foránea con la tabla carreras. Esta relación asegura que las generaciones se asocien correctamente a una carrera, y la eliminación en cascada ayuda a mantener la integridad de los datos si una carrera se elimina.*

```

<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateGeneracionesTable extends Migration
{
    public function up()
    {
        Schema::create('generaciones', function (Blueprint $table) {
            $table->id('GeneracionID');
            $table->string('Nombre');
            $table->foreignId('CarreraID') ->constrained('carreras',
            'CarreraID')
            ->onDelete('cascade');
            $table->timestamps();
        });
    }

    public function down()
    {
        Schema::dropIfExists('generaciones');
    }
}

```

Anexo 10. Migración de la tabla generaciones

Tabla grupos

*La tabla grupos permite organizar a los estudiantes en equipos específicos dentro de una generación académica. Cada grupo está identificado por un **Nombre** y se relaciona con la tabla generaciones mediante una clave foránea. Esto asegura que cada grupo pertenezca a una generación definida, garantizando la coherencia en la jerarquía académica. La eliminación en cascada facilita la limpieza de datos relacionados cuando una generación es removida.*

```

<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateGruposTable extends Migration
{
    public function up()
    {
        Schema::create('grupos', function (Blueprint $table) {
            $table->id('GrupoID');
            $table->string('Nombre');
            $table->foreignId('GeneracionID') ->constrained('generaciones', 'GeneracionID')
                ->onDelete('cascade');
            $table->timestamps();
        });
    }

    public function down()
    {
        Schema::dropIfExists('grupos');
    }
}

```

Anexo 11. Migración de la tabla grupos

Tabla cuestionarios

*La tabla cuestionarios está diseñada para gestionar evaluaciones asignadas a los grupos de estudiantes. Cada cuestionario incluye detalles como **Nombre**, **Descripción**, **fechas de publicación y cierre**, y está vinculado a la tabla grupos mediante una relación foránea con eliminación en cascada. Este diseño facilita la administración de evaluaciones específicas para cada grupo, asegurando integridad en la gestión académica.*

```

<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateCuestionariosTable extends Migration
{
    public function up()
    {
        Schema::create('cuestionarios', function (Blueprint $table) {
            $table->id('CuestionarioID');
            $table->foreignId('GrupoID')->nullable()->constrained('grupos', 'GrupoID')
                ->onDelete('cascade');
            $table->string('Nombre');
            $table->text('Descripcion')->nullable();
            $table->timestamps();
        });
    }

    public function down()
    {
        Schema::dropIfExists('cuestionarios');
    }
}

```

Anexo 12. Migración de la tabla Cuestionarios

Tabla preguntas

La tabla preguntas organiza las preguntas asociadas a un cuestionario específico. Cada registro incluye el texto de la pregunta, su tipo (positiva o negativa) y una relación foránea con la tabla cuestionarios, lo que permite una eliminación en cascada. Este esquema garantiza una estructura clara para gestionar las evaluaciones dentro de la plataforma.

```

<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreatePreguntasTable extends Migration
{
    public function up()
    {
        Schema::create('preguntas', function (Blueprint $table) {
            $table->id('PreguntaID');
            $table->text('Texto');
            $table->enum('Tipo', ['Positiva', 'Negativa']);
            $table->foreignId('CuestionarioID') ->constrained('cuestionarios', 'CuestionarioID')
            ->onDelete('cascade');
            $table->timestamps();
        });
    }

    public function down()
    {
        Schema::dropIfExists('preguntas');
    }
}

```

Anexo 13. Migración de la tabla Preguntas

Tabla respuestas

La tabla respuestas almacena las opciones asociadas a cada pregunta en un cuestionario. Incluye el texto de la respuesta, un valor numérico que refleja su puntuación y una relación foránea con la tabla preguntas, lo que asegura consistencia y eliminación en cascada. Esta estructura facilita la evaluación precisa y organizada de las respuestas dentro del sistema.

```

<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateRespuestasTable extends Migration
{
    public function up()
    {
        Schema::create('respuestas', function (Blueprint $table) {
            $table->id('RespuestaID');
            $table->foreignId('PreguntaID') ->constrained('preguntas', 'PreguntaID')
                ->onDelete('cascade');
            $table->text('Texto');
            $table->tinyInteger('Valor');
            $table->timestamps();
        });
    }

    public function down()
    {
        Schema::dropIfExists('respuestas');
    }
}

```

Anexo 14. Migración de la tabla respuestas

Tabla usuarios

La tabla usuarios almacena los datos personales y académicos de los usuarios, incluyendo su nombre, matrícula, semestre, correo electrónico y detalles como edad, género, y credenciales de acceso. Además, se establece una relación con las tablas instituciones, países y carreras a través de claves foráneas, permitiendo una estructura flexible y consistente. Esta tabla es crucial para gestionar los diferentes tipos de usuarios en el sistema (alumno, maestro, admin) y sus interacciones con otras entidades en la plataforma.

```

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateUsuariosTable extends Migration
{
    public function up()
    {
        Schema::create('usuarios', function (Blueprint $table) {
            $table->id('UsuarioID');
            $table->string('Nombre');
            $table->string('Primer_apellido');
            $table->string('Segundo_apellido')->nullable();
            $table->string('Matricula')->unique();
            $table->string('Semestre');
            $table->enum('Tipo', ['Alumno', 'Maestro', 'Admin']);
            $table->string('Correo')->unique();
            $table->string('Password');
            $table->integer('edad')->nullable();
            $table->enum('genero', ['Masculino', 'Femenino', 'Otro'])->nullable();
            $table->unsignedBigInteger('InstitucionID')->nullable();
            $table->unsignedBigInteger('IDPais')->nullable();
            $table->unsignedBigInteger('CarreraID')->nullable();
            $table->timestamps();
            $table->foreign('InstitucionID')->references('InstitucionID')->on('instituciones')->onDelete('set
null');
            $table->foreign('IDPais')->references('IDPais')->on('paises')->onDelete('set null');
            $table->foreign('CarreraID')->references('CarreraID')->on('carreras')->onDelete('set null');
        });
    }

    public function down()
    {
        Schema::dropIfExists('usuarios');
    }
}

```

Anexo 15. Migración de la tabla usuarios

Tabla intermedia para la relación entre el cuestionario y grupo

La tabla cuestionario_grupo establece una relación de muchos a muchos entre los cuestionarios y los grupos. Cada registro asocia un cuestionario específico a un grupo determinado, permitiendo que los cuestionarios sean asignados a diferentes grupos de estudiantes. Esta migración incluye claves foráneas que vinculan las tablas cuestionarios y grupos, asegurando la integridad referencial y facilitando la gestión de los cuestionarios dentro de los grupos correspondientes.

```
<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateCuestionarioGrupoTable extends Migration
{
    public function up()
    {
        Schema::create('cuestionario_grupo', function (Blueprint $table) {
            $table->id('CuestionarioGrupoID');
            $table->foreignId('CuestionarioID') ->constrained('cuestionarios', 'CuestionarioID')
                ->onDelete('cascade');
            $table->foreignId('GrupoID') ->constrained('grupos', 'GrupoID')
                ->onDelete('cascade');
            $table->date('FechaPublicacion');
            $table->date('FechaCierre');
            $table->timestamps();
        });
    }

    public function down()
    {
        Schema::dropIfExists('cuestionario_grupo');
    }
}
```

Anexo 16. Migración de la tabla cuestionario grupo

Tabla intermedia para la relación del usuario con el grupo

La tabla grupo_usuario establece una relación de muchos a muchos entre los grupos y los usuarios (alumnos, maestros, o administradores). Cada registro en esta tabla asocia un usuario

específico con un grupo determinado, permitiendo que los usuarios pertenezcan a varios grupos y viceversa. Esta migración utiliza claves foráneas para vincular las tablas grupos y usuarios, asegurando que los datos estén correctamente referenciados y manteniendo la integridad de la base de datos.

```
<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateGrupoUsuarioTable extends Migration
{ public function up()
    { Schema::create('grupo_usuario', function (Blueprint $table) {
        $table->id();
        $table->foreignId('GrupoID')->constrained('grupos', 'GrupoID');
        $table->foreignId('UsuarioID')->constrained('usuarios', 'UsuarioID');
        $table->timestamps(); }); }

    public function down()
    { Schema::dropIfExists('grupo_usuario'); } }
```

Anexo 17. Migración de la tabla grupo con usuario

Tabla categorías

La tabla *categorias* se encarga de almacenar las categorías que pueden ser asignadas a las preguntas de los cuestionarios. Esta migración crea una tabla sencilla con un campo **CategoriaID** como clave primaria y un campo *Nombre* para almacenar el nombre de la categoría. Al ser una tabla básica, se utiliza para organizar las preguntas en diferentes categorías, lo que facilita su clasificación y búsqueda. Además, se incluye la gestión automática de las marcas de tiempo (**created_at** y **updated_at**).

```

<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateCategoriasTable extends Migration
{
    public function up()
    {
        Schema::create('categorias', function (Blueprint $table) {
            $table->id('CategorialID');
            $table->string('Nombre');
            $table->timestamps();
        });
    }

    public function down()
    {
        Schema::dropIfExists('categorias');
    }
}

```

Anexo 18. Migración tabla categorías

Tablas variables

La tabla variable tiene como propósito almacenar las variables asociadas a las categorías de preguntas. Esta migración crea un identificador único VariableID para cada variable, un campo Nombre para describir la variable, y un campo CategoriaID que establece una relación con la tabla categorias. Esta relación se asegura de que, al eliminar una categoría, se eliminen también las variables asociadas a ella, utilizando la opción onDelete('cascade').

```

<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateVariablesTable extends Migration
{
    public function up()
    {
        Schema::create('variables', function (Blueprint $table) {
            $table->id('VariableID');
            $table->string('Nombre');
            $table->foreignId('CategorialID') ->constrained('categorias', 'CategorialID')
                ->onDelete('cascade');
            $table->timestamps();
        });
    }

    public function down()
    {
        Schema::dropIfExists('variables');
    }
}

```

Anexo 19. Migración tabla variables

Tabla intermedia para la relación entre las variables y las preguntas

La migración pregunta_variable crea una tabla intermedia que establece una relación de muchos a muchos entre las tablas preguntas y variables. Cada registro en esta tabla representa una asignación de una variable a una pregunta específica, identificada por el campo PreguntaVariableID. La relación entre las preguntas y las variables se define mediante las claves foráneas PreguntaID y VariableID, que están configuradas para eliminar en cascada si la pregunta o la variable asociada se eliminan.

```

<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreatePreguntaVariableTable extends Migration
{
    public function up()
    {
        Schema::create('pregunta_variable', function (Blueprint $table) {
            $table->id('PreguntaVariableID');
            $table->foreignId('PreguntaID') ->constrained('preguntas', 'PreguntaID')
                ->onDelete('cascade');
            $table->foreignId('VariableID') ->constrained('variables', 'VariableID')
                ->onDelete('cascade');
            $table->timestamps();
        });
    }

    public function down()
    {
        Schema::dropIfExists('pregunta_variable');
    }
}

```

Anexo 20. Migración tabla Pregunta Variable

Tabla respuestas enviadas

*La tabla respuestas enviadas se encarga de almacenar las respuestas de los cuestionarios contestados por el usuario. Esta migración crea una tabla sencilla con un campo **RespuestaEnviadaID** como clave primaria y establece una relación entre la tabla cuestionarios, usuarios y preguntas que en este caso se utiliza igual el valor de la respuesta. Además, se incluye la gestión automática de las marcas de tiempo (**created_at** y **updated_at**).*

```

<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateRespuestasEnviadasTable extends Migration
{
    public function up(): void
    {
        Schema::create('respuestas_enviadas', function (Blueprint $table) {
            $table->id('RespuestaEnviadaID');
            $table->foreignId('CuestionarioID') ->constrained('cuestionarios', 'CuestionarioID')
                ->onDelete('cascade');
            $table->foreignId('PreguntaID') ->constrained('preguntas', 'PreguntaID')
                ->onDelete('cascade');
            $table->integer('respuesta_valor');
            $table->foreignId('UsuarioID')->constrained('usuarios', 'UsuarioID')
                ->onDelete('cascade');
            $table->timestamps(); });
    }

    public function down(): void
    {
        Schema::dropIfExists('respuestas_enviadas');
    }
}

```

Anexo 21. Migración tabla respuestas enviadas

Tabla expresiones

*La migración expresiones define una tabla que relaciona las expresiones con las variables. Cada expresión tiene un identificador único *ExpresionesID*, un campo de texto *Nombre* para su descripción, y una clave foránea *VariableID* que vincula la expresión con una variable específica. La relación está configurada para eliminar las expresiones en cascada si la variable asociada se elimina, asegurando la integridad referencial en la base de datos.*

```

<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateExpresionesTable extends Migration
{
    public function up()
    {
        Schema::create('expresiones', function (Blueprint $table) {
            $table->id('ExpresionesID');
            $table->string('Nombre');
            $table->foreignId('VariableID') ->constrained('variables', 'VariableID')
                ->onDelete('cascade');
            $table->timestamps();
        });
    }

    public function down(): void
    {
        Schema::dropIfExists('expresiones');
    }
}

```

Anexo 1 Migración de la tabla expresiones

Tabla cuestionarios resueltos

La migración cuestionario_resuelto crea una tabla para registrar los cuestionarios completados por los usuarios, asociándolos a un grupo y cuestionario específico. Incluye claves foráneas a usuarios, cuestionarios y grupos, con eliminación en cascada para mantener la integridad referencial. Además, almacena la fecha de resolución y un indicador booleano para marcar si el cuestionario ha sido resuelto.

```

<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateCuestionarioResueltoTable extends Migration
{
    public function up()
    {
        Schema::create('cuestionario_resuelto', function (Blueprint $table) {
            $table->id('CuestionarioResueltoID');
            $table->foreignId('UsuarioID') ->constrained('usuarios', 'UsuarioID')
                ->onDelete('cascade');
            $table->foreignId('CuestionarioID') ->constrained('cuestionarios', 'CuestionarioID')
                ->onDelete('cascade');
            $table->foreignId('GrupoID') ->constrained('grupos', 'GrupoID')
                ->onDelete('cascade');
            $table->timestamp('FechaResuelto');
            $table->boolean('Resuelto')->default(false);
            $table->timestamps(); });
    }

    public function down()
    {
        Schema::dropIfExists('cuestionario_resuelto');
    }
}

```

Anexo 2 Migración de la tabla cuestionario resuelto

Tabla intermedia que relaciona el cuestionario con las preguntas

La migración ***cuestionario_pregunta*** crea una tabla intermedia que establece una relación de uno a muchos entre las tablas ***cuestionarios*** y ***preguntas***. Cada registro en esta tabla representa una asignación de un cuestionario a un conjunto de preguntas seleccionadas, identificada por el campo ***CuestionarioPreguntaID***. La relación entre el cuestionario y las preguntas se define mediante las claves foráneas ***CuestionarioID*** y ***PreguntaID***, que están configuradas para eliminar en cascada si el cuestionario o la pregunta asociada se eliminan.

```

<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateCuestionarioPreguntaTable extends Migration
{
    public function up()
    {
        Schema::create('cuestionario_pregunta', function (Blueprint $table) {
            $table->id('CuestionarioPreguntaID');
            $table->foreignId('CuestionarioID') ->constrained('cuestionarios', 'CuestionarioID')
                ->onDelete('cascade');
            $table->foreignId('PreguntaID') ->constrained('preguntas', 'PreguntaID')
                ->onDelete('cascade');
            $table->timestamps();
        });
    }

    public function down()
    {
        Schema::dropIfExists('cuestionario_pregunta');
    }
}

```

Anexo 3 Migración de la tabla cuestionario pregunta

Tabla intermedia que relaciona las categorías con las expresiones

La migración ***categorias_expresion*** crea una tabla intermedia para establecer una relación muchos a muchos entre las tablas ***categorias*** y ***expresiones***. Define claves foráneas hacia ***categorias*** y ***expresiones***, asegurando la eliminación en cascada para mantener la integridad referencial. Además, incluye un identificador único (***CategoriaExpresionID***) y marcas de tiempo (***timestamps***) para seguimiento temporal.


```

<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateCategoriasExpresionTable extends Migration
{
    public function up()
    {
        Schema::create('categorias_expresion', function (Blueprint $table) {
            $table->id('CategoriaExpresionID');
            $table->foreignId('CategorialD') ->constrained('categorias', 'CategorialD')
                ->onDelete('cascade');
            $table->foreignId('ExpresionesID') ->constrained('expresiones', 'ExpresionesID')
                ->onDelete('cascade');
            $table->timestamps();
        });
    }

    public function down()
    {
        Schema::dropIfExists('categorias_expresion');
    }
}

```

Anexo 4 Migración de la tabla categorías expresiones

Tabla puntajes por variable

La migración puntajes_por_variable crea una tabla para almacenar los puntajes obtenidos por los usuarios en cada variable asociada a un cuestionario dentro de un grupo. Incluye columnas para identificar al usuario, grupo, cuestionario y variable, así como el puntaje correspondiente. Las claves foráneas hacia las tablas usuarios, grupos, cuestionarios y variables garantizan la integridad referencial, y se eliminan en cascada para mantener consistencia en los datos.

```

<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreatePuntajesVariablesTable extends Migration
{
    public function up()
    {
        Schema::create('puntajes_por_variable', function (Blueprint $table) {
            $table->id('PuntajeID');
            $table->unsignedBigInteger('UsuarioID');
            $table->unsignedBigInteger('GrupoID');
            $table->unsignedBigInteger('CuestionarioID');
            $table->unsignedBigInteger('VariableID');
            $table->integer('puntaje')->nullable();
            $table->timestamps();
            $table->foreign('UsuarioID')->references('UsuarioID') ->on('usuarios')
            ->onDelete('cascade');
            $table->foreign('GrupoID')->references('GrupoID') ->on('grupos')
            ->onDelete('cascade');
            $table->foreign('CuestionarioID')->references ('CuestionarioID')
            ->on('cuestionarios')->onDelete('cascade');
            $table->foreign('VariableID')->references('VariableID')->on('variables')
            ->onDelete('cascade'); });
    }

    public function down()
    {
        Schema::dropIfExists('puntajes_por_variable');
    }
}

```

Anexo 5 Migración de la tabla puntajes por variables

Tabla sumatoria puntuaciones

La migración sumatoria_puntuaciones crea una tabla para registrar las sumatorias y porcentajes relacionados con las puntuaciones de un usuario en un cuestionario. Incluye referencias a categorías, variables y expresiones relacionadas, además del puntaje máximo, la sumatoria total, y el porcentaje calculado para cada combinación. Todas las relaciones están respaldadas por claves foráneas para garantizar la integridad de los datos y eliminar registros relacionados en cascada. Los campos numéricos como total_puntuacion y porcentaje están diseñados con alta precisión para almacenar resultados exactos.

```

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateSumatoriaPuntuacionesTable extends Migration
{
    public function up()
    {
        Schema::create('sumatoria_puntuaciones', function (Blueprint $table) {
            $table->id('SumatorialID');
            $table->unsignedBigInteger('Usuarioid');
            $table->unsignedBigInteger('CategorialID');
            $table->unsignedBigInteger('VariableID');
            $table->unsignedBigInteger('ExpresionesID');
            $table->unsignedBigInteger('Cuestionarioid');
            $table->decimal('total_puntuacion', 8, 2)->default(0);
            $table->decimal('puntaje_maximo', 8, 2)->default(0);
            $table->decimal('porcentaje', 5, 2)->default(0);
            $table->timestamps();
            $table->foreign('Usuarioid')->references('Usuarioid') ->on('usuarios')
                ->onDelete('cascade');
            $table->foreign('CategorialID')->references('CategorialID') ->on('categorias')
                ->onDelete('cascade');
            $table->foreign('VariableID')->references('VariableID') ->on('variables')
                ->onDelete('cascade');
            $table->foreign('ExpresionesID')->references('ExpresionesID') ->on('expresiones')
                ->onDelete('cascade');
            $table->foreign('Cuestionarioid')->references('Cuestionarioid')->on('cuestionarios')
                ->onDelete('cascade'); });
    }

    public function down()
    {
        Schema::dropIfExists('sumatoria_puntuaciones');
    }
}

```

Modelos

En este proyecto los modelos representan la estructura de las tablas en la base de datos y permiten interactuar con ellas de manera sencilla mediante Eloquent, el ORM que facilita consultas, relaciones y operaciones CRUD. Cada modelo refleja las entidades del sistema, como usuarios, grupos, instituciones, países, carreras, categorías, expresiones, variables, cuestionarios y respuestas, permitiendo establecer relaciones como "uno a muchos" o "muchos a muchos". Estas definiciones son clave para mantener la lógica y la integridad de los datos del proyecto, alineándolas con las migraciones que estructuran la base de datos.

Modelos usuarios

El modelo Usuario gestiona los datos de los usuarios de la plataforma y está relacionado con entidades como grupos, carreras, instituciones y países. Incluye métodos para relaciones muchos a muchos y claves foráneas, ocultando campos sensibles y permitiendo la autenticación mediante Authenticatable. Es clave para conectar usuarios con otras funcionalidades del sistema.

```
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Laravel\Sanctum\HasApiTokens;
use App\Models\Grupos;

class Usuario extends Authenticatable
{
    use HasApiTokens, HasFactory;

    protected $table = 'usuarios';
    protected $primaryKey = 'UsuarioID';
    protected $fillable = [
        'Nombre', 'Primer_apellido', 'Segundo_apellido', 'Matricula', 'Semestre', 'Tipo',
        'Correo', 'Password', 'edad', 'genero', 'InstitucionID', 'IDPais', 'CarreraID'];
}
```

```

protected $hidden = [ 'Password', 'remember_token', ];

// Casts para tipos de datos
protected $casts = [ 'Correo_verified_at' => 'datetime', ];

// Relación muchos a muchos con grupos
public function grupos()
{ return $this->belongsToMany(Grupos::class, 'grupo_usuario', 'UsuariID', 'GrupID')
  ->withPivot('CarreraID', 'InstitucionID') ->withTimestamps(); }

// Relación con la carrera
public function carrera()
{ return $this->belongsTo(Carreras::class, 'CarreraID'); }

// Relación con la institución
public function institucion()
{ return $this->belongsTo(Instituciones::class, 'InstitucionID'); }

// Relación con el país
public function pais()
{return $this->belongsTo(Paises::class, 'IDPais'); } }

```

Anexo 8 Relaciones del modelo usuarios

Modelo países

El modelo Paises gestiona la tabla paises y su relación con los usuarios de la plataforma. Define atributos básicos como el nombre y establece una relación uno a muchos con el modelo Usuario, conectando cada país con sus usuarios asociados. Incluye configuraciones clave como el uso de una clave primaria personalizada (IDPais).

```

<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Model;

class Paises extends Model
{
    protected $table = 'paises';
    protected $primaryKey = 'IDPais';
    protected $fillable = ['Nombre'];
    public $incrementing = true;
    protected $keyType = 'int';
    public $timestamps = true;

    // Relación con Usuarios
    public function usuarios()
    {
        return $this->hasMany(Usuario::class, 'IDPais', 'IDPais');
    }
}

```

Anexo 9 Modelo países

Modelo instituciones

El modelo Instituciones representa la tabla instituciones y las relaciones con otros modelos como Usuario, Carreras y Paises. Define los atributos Nombre y IDPais, y establece configuraciones para la clave primaria, utilizando InstitucionID como clave primaria. El modelo también contiene varias relaciones: tiene muchos usuarios (usuarios), muchas carreras (carreras), y pertenece a un país específico (pais).

```

<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Model;

class Instituciones extends Model
{
    protected $table = 'instituciones';
    protected $primaryKey = 'InstitucionID';
    protected $fillable = ['Nombre', 'IDPais'];
    public $incrementing = true;
    protected $keyType = 'int';
    public $timestamps = true;
}

```

Anexo 10 Modelo Instituciones

```

// Relación con Usuarios
public function usuarios()
{
    return $this->hasMany(Usuario::class, 'InstitucionID', 'InstitucionID');
}

// Relación con Carreras
public function carreras()
{
    return $this->hasMany(Carreras::class, 'InstitucionID', 'InstitucionID');
}

// Relación con Países
public function pais()
{
    return $this->belongsTo(Paises::class, 'IDPais', 'IDPais');
}
}

```

Anexo 11 Relaciones del modelo Instituciones

Modelo carreras

Este modelo nos permitio gestionar la información de las carreras dentro de una institución educativa y relacionarla con los usuarios que pertenecen a ellas. El modelo Carreras representa la tabla carreras y contiene las relaciones con Usuarios e Instituciones. Define los atributos

Nombre e InstitucionID, y tiene configurada la clave primaria CarreraID. Además, establece que una carrera puede tener muchos usuarios asociados (usuarios), y cada carrera pertenece a una institución específica (institucion).

```
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Carreras extends Model
{ use HasFactory;
    protected $table = 'carreras';
    protected $primaryKey = 'CarreraID';
    protected $fillable = [ 'Nombre', 'InstitucionID', ];
    public $incrementing = true;
    protected $keyType = 'int';
    public $timestamps = true;

    // Definir la relación con Usuarios
    public function usuarios()
    { return $this->hasMany(Usuario::class, 'CarreraID', 'CarreraID');}

    // Definir la relación con Institución
    public function institucion()
    { return $this->belongsTo(Instituciones::class, 'InstitucionID', 'InstitucionID'); } }
```

Anexo 12 Modelo Carreras

Modelo carrera grupos

El modelo CarreraGrupos representa la tabla carreras y define una relación con el modelo Instituciones. La clave primaria en este modelo es CarreraID, y la relación belongsTo establece que cada carrera pertenece a una institución específica. Aquí asociamos los grupos o carreras a una institución educativa, que nos facilita la gestión de los datos de las carreras dentro una institución.

```
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use App\Models\Instituciones;

class CarreraGrupos extends Model
{ use HasFactory;

    protected $table = 'carreras';
    protected $primaryKey = 'CarreraID';

    //Relacion con las instituciones
    public function institucion()
    { return $this->belongsTo(Instituciones::class, 'InstitucionID'); } }
```

Anexo 28 Modelo carrera grupos

Modelo usuario con grupos

El modelo UsuarioGrupo nos permitió asociar a los usuarios con los grupos a los que están asignados, mediante la tabla pivote grupo_usuario. Esto facilitó la inclusión de detalles adicionales como la carrera y la institución del usuario dentro de la relación. Además, definimos una relación de pertenencia con el modelo Grupos, lo que optimizó la gestión y consulta de estos vínculos en la plataforma.

```

<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use App\Models\Grupos;

class UsuarioGrupo extends Model
{
    use HasFactory;

    protected $table = 'grupo_usuario';
    protected $fillable = [ 'UsuarioID', 'GrupoID', 'CarreraID', 'InstitucionID', ];

    // Relación con grupos (tabla pivote grupo_usuario)
    public function grupo()
    {
        return $this->belongsTo(Grupos::class, 'GrupoID');
    }
}

```

Anexo 29 Modelo usuario grupo

Modelo cuestionario

El modelo Cuestionario nos permitió asociar cuestionarios con grupos específicos, lo que facilita su gestión dentro de la plataforma. A través de la relación con el modelo Grupos, podemos asignar cada cuestionario a un grupo determinado. Además, logramos vincular las preguntas de cada cuestionario mediante la tabla pivote cuestionario_pregunta, lo que nos permite organizar y relacionar de manera flexible las preguntas con los cuestionarios. También se incluye una relación con CuestionarioGrupo, lo que permite gestionar los grupos que participan en un cuestionario.

```

<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use App\Models\Grupos;

class Cuestionario extends Model
{ use HasFactory;
    protected $primaryKey = 'CuestionarioID';
    protected $fillable = [ 'GrupoID', 'Nombre', 'Descripcion', ];

    //Relacion con los grupos
    public function grupo()
    { return $this->belongsTo(Grupos::class, 'GrupoID'); }

    //Relacion con las preguntas (tabla cuestionario_pregunta)
    public function preguntas()
    { return $this->belongsToMany(Pregunta::class, 'cuestionario_pregunta',
        'CuestionarioID', 'PreguntaID') ->with('respuestas'); }

    public function cuestionarioGrupos()
    { return $this->hasMany(CuestionarioGrupo::class, 'CuestionarioID'); } }

```

Anexo 30 Modelo cuestionario

Modelo cuestionario con relación al grupo

El modelo CuestionarioGrupo nos permitió asociar un cuestionario con un grupo específico dentro de la plataforma. Mediante la relación con el modelo Cuestionario, podemos vincular un cuestionario con su correspondiente grupo, lo que facilita la gestión de los cuestionarios para cada grupo. También implementamos las fechas de publicación y cierre del cuestionario, permitiendo que solo se pueda acceder a él dentro de ese rango de tiempo. A través de la relación con Grupos, aseguramos que cada grupo tenga su propio conjunto de cuestionarios asignados.

```

<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use App\Models\Cuestionario;
use App\Models\Grupos;

class CuestionarioGrupo extends Model
{ use HasFactory;
    protected $table = 'cuestionario_grupo';
    protected $primaryKey = 'CuestionarioGrupoID';
    protected $fillable = [ 'CuestionarioID', 'GrupoID', 'FechaPublicacion', 'FechaCierre', ];

    // Relación con el cuestionario
    public function cuestionario()
    { return $this->belongsTo(Cuestionario::class, 'CuestionarioID'); }

    // Relación con los grupos
    public function grupo()
    { return $this->belongsTo(Grupos::class, 'GrupoID'); } }

```

Anexo 31 Modelo cuestionario grupo

Modelo cuestionario con relación pregunta

Este modelo CuestionarioPregunta nos permitió asociar los cuestionarios con las preguntas específicas a través de una tabla intermedia. Esta relación muchos a muchos facilitó la asignación de varias preguntas a un cuestionario, manteniendo la flexibilidad de estructuración en los cuestionarios sin perder el control sobre las preguntas que se asignan a cada uno.

```

<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class CuestionarioPregunta extends Model
{
    protected $table = 'cuestionario_pregunta';
    protected $primaryKey = 'CuestionarioPreguntaID';
    use HasFactory;
    protected $fillable = [ 'CuestionarioID', 'PreguntaID', ];

    // Relación con el cuestionario
    public function cuestionario()
    {
        return $this->belongsTo(Cuestionario::class, 'CuestionarioID');
    }

    // Relación con la pregunta
    public function pregunta()
    {
        return $this->belongsTo(Pregunta::class, 'PreguntaID', 'PreguntaID');
    }
}

```

Anexo 32 Modelo cuestionario pregunta

Modelo cuestionario resuelto

El modelo CuestionarioResuelto nos permite asociar los cuestionarios resueltos con los usuarios y los grupos correspondientes, registrando la fecha en que se resolvió cada cuestionario. Esto nos facilitó mantener un historial de los cuestionarios completados por cada usuario dentro de los grupos específicos. Además, la relación con los modelos Usuario, Cuestionario y Grupos asegura que los datos estén correctamente vinculados y se pueda acceder fácilmente a la información relevante para cada respuesta registrada.

```

<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class CuestionarioResuelto extends Model
{ use HasFactory;
    protected $table = 'cuestionario_resuelto';
    protected $primaryKey = 'CuestionarioResueltoID';
    protected $fillable = [ 'UsuarioID', 'CuestionarioID', 'GrupoID', 'FechaResuelto',
        'Resuelto' ];

    // Relación con el usuario
    public function usuario()
    { return $this->belongsTo(Usuario::class, 'UsuarioID'); }

    // Relación con el cuestionario
    public function cuestionario()
    { return $this->belongsTo(Cuestionario::class, 'CuestionarioID'); }

    // Relación con el grupo
    public function grupo()
    { return $this->belongsTo(Grupos::class, 'GrupoID'); } }

```

Anexo 33 Modelo cuestionario resuelto

Modelo expresiones

El modelo Expresiones nos permitió asociar cada expresión con su respectiva variable mediante la relación con el modelo Variable. Esto facilita el manejo de datos, ya que cada expresión se vincula correctamente a su variable correspondiente, haciendo más eficiente el procesamiento de

la información. Además, su uso en el sistema permite organizar y acceder a las expresiones relacionadas de manera sencilla.

```
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Expresiones extends Model
{ use HasFactory;
    protected $table = 'expresiones';
    protected $primaryKey = 'ExpresionesID';
    protected $fillable = ['Nombre', 'VariableID'];
    public $timestamps = true;

    // Relación con la variable
    public function variable()
    { return $this->belongsTo(Variable::class, 'VariableID'); } }
```

Anexo 34 Modelo expresiones

Modelo generaciones

El modelo Generaciones nos permitió asociar cada generación con una carrera específica mediante la relación con el modelo Carreras. Esto facilita la gestión de las generaciones dentro de un programa académico, permitiendo acceder fácilmente a las generaciones por carrera. La relación con el modelo Carreras también permite organizar y filtrar generaciones de manera más eficiente.


```
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use App\Models\Carreras;
class Generaciones extends Model
{ use HasFactory;
    protected $table = 'generaciones';
    protected $primaryKey = 'GeneracionID';
    protected $fillable = [ 'Nombre', 'CarreraID', ];

    // Relación con Carrera
    public function carrera()
    { return $this->belongsTo(Carreras::class, 'CarreraID'); } }
```

Anexo 35 Modelo generaciones

Modelo grupos

El modelo Grupos nos permitió asociar grupos con carreras, generaciones y usuarios, facilitando la gestión de la pertenencia de los estudiantes a grupos específicos. Además, implementamos un sistema para generar un código de acceso único para cada grupo antes de su creación, lo que agiliza el proceso de inscripción. La relación con Usuario y CarreraGrupos asegura una organización eficiente dentro del sistema.

```

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use App\Models\CarreraGrupos;
use App\Models\Usuario;
use Illuminate\Support\Str;

class Grupos extends Model
{ use HasFactory;

    protected $table = 'grupos';
    protected $primaryKey = 'GrupoID';
    protected $fillable = ['Nombre', 'codigo_acceso', 'semestre', 'GeneracionID',
    'CarreraID', 'UsuarioID'];

    // Relación con carreras
    public function carrera()
    { return $this->belongsTo(CarreraGrupos::class, 'CarreraID')->with('institucion'); }

    // Relación con usuarios (muchos a muchos)
    public function usuarios()
    { return $this->belongsToMany(Usuario::class, 'grupo_usuario', 'GrupoID',
    'UsuarioID')->withPivot('CarreraID'); }

    // Método boot para añadir el evento creating
    protected static function boot()
    { parent::boot();

        static::creating(function ($grupo) { $grupo->codigo_acceso = Str::random(8); }); }
    public function usuario()
    { return $this->belongsTo(Usuario::class, 'UsuarioID'); } }

```

Modelo pregunta

El modelo Pregunta nos permitió asociar preguntas con múltiples respuestas y variables, facilitando la gestión de las preguntas dentro de los cuestionarios. A través de las relaciones con Respuesta y Variable, logramos una estructura flexible para almacenar las respuestas y vincularlas a diferentes variables, lo que facilita la interpretación y análisis de los datos recolectados.

```
<?php
namespace App\Models;
use App\Models\Respuesta;
use App\Models\Variable;
use Illuminate\Database\Eloquent\Model;

class Pregunta extends Model
{ protected $table = 'preguntas';
    protected $primaryKey = 'PreguntaID';
    public $timestamps = true;
    protected $fillable = ['PreguntaID', 'Texto', 'Tipo'];

    // Relación con las Respuesta
    public function respuestas()
    { return $this->hasMany(Respuesta::class, 'PreguntaID', 'PreguntaID'); }

    // Relación con las variables
    public function variables()
    { return $this->belongsToMany(Variable::class, 'pregunta_variable', 'PreguntaID',
    'VariableID'); } }
```

Anexo 37 Modelo pregunta

Modelo puntajes por variable

El modelo PuntajesPorVariable nos permite asociar el puntaje obtenido por un usuario en un cuestionario con una variable específica dentro de un grupo. A través de las relaciones con

Usuario, Grupo, Cuestionario y Variable, logramos almacenar y gestionar los puntajes de manera eficiente, facilitando la interpretación de los resultados por grupo y variable

```
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class PuntajesPorVariable extends Model
{ use HasFactory;
    protected $table = 'puntajes_por_variable';
    protected $primaryKey = 'PuntajeID';
    protected $fillable = [ 'UsuarioID', 'GrupoID', 'CuestionarioID', 'VariableID', 'puntaje', ];

    public function usuario()
    { return $this->belongsTo(User::class, 'UsuarioID'); }

    public function grupo()
    { return $this->belongsTo(Grupos::class, 'GrupoID'); }

    public function cuestionario()
    { return $this->belongsTo(Cuestionario::class, 'CuestionarioID'); }

    public function variable()
    { return $this->belongsTo(Variable::class, 'VariableID'); } }
```

Anexo 38 Modelo puntajes por variable

Modelo respuesta enviada

El modelo RespuestaEnviada nos permite asociar las respuestas de un usuario a un cuestionario específico, guardando el valor de la respuesta junto con la pregunta correspondiente. La relación con los modelos Pregunta, Usuario y Cuestionario permite rastrear qué respuesta se ha enviado

por cada usuario en un cuestionario determinado. Esto facilita la administración de los datos y la generación de informes sobre el desempeño de los estudiantes en cada cuestionario.

```
<?php
namespace App\Models;
use App\Models\Pregunta;
use App\Models\Usuario;
use App\Models\Cuestionario;

use Illuminate\Database\Eloquent\Model;

class RespuestaEnviada extends Model
{ protected $table = 'respuestas_enviadas';
  protected $primaryKey = 'RespuestaEnviadaID';
  protected $fillable = [ 'CuestionarioID', 'PreguntaID', 'respuesta_valor', 'UsuarioID', ];

  // Relación con la tabla de Preguntas
  public function pregunta()
  { return $this->belongsTo(Pregunta::class, 'PreguntaID'); }

  // Relación con la tabla de Usuario (Alumno)
  public function usuario()
  { return $this->belongsTo(Usuario::class, 'UsuarioID'); }

  // Relación con la tabla de Cuestionarios
  public function cuestionario()
  { return $this->belongsTo(Cuestionario::class, 'CuestionarioID'); } }
```

Modelo sumatoria puntajes

El modelo SumatoriaPuntajes permite almacenar la puntuación total obtenida por un usuario en relación con un cuestionario, categoría, variable y expresión específicos. Este modelo asocia el puntaje total con variables como el puntaje máximo y el porcentaje obtenido, lo que facilita el cálculo de los resultados y el análisis del desempeño. Las relaciones con los modelos Usuario, Variable, Cuestionario, Categoría y Expresiones permiten un seguimiento detallado de las evaluaciones de los estudiantes en función de diferentes criterios.

```
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class SumatoriaPuntajes extends Model
{ use HasFactory;
    protected $table = 'sumatoria_puntuaciones';
    protected $primaryKey = 'SumatorialD';
    protected $fillable = [ 'UsuarioID', 'CategorialD', 'VariableID', 'ExpresionesID',
'CuestionarioID', 'total_puntuacion', 'puntaje_maximo', 'porcentaje' ];

    //Relaciones con otros modelos
    public function usuario()
    { return $this->belongsTo(Usuario::class, 'UsuarioID'); }
    public function variable()
    { return $this->belongsTo(Variable::class, 'VariableID'); }
    public function cuestionario()
    { return $this->belongsTo(Cuestionario::class, 'CuestionarioID'); }
    public function categoria()
    { return $this->belongsTo(Categoria::class, 'CategorialD'); }
    public function expresion()
    { return $this->belongsTo(Expresiones::class, 'ExpresionesID'); } }
```

Modelo variable

El modelo Variable está diseñado para gestionar las variables que se utilizan en los cuestionarios, vinculándolas con categorías y preguntas específicas. Las relaciones con los modelos Pregunta, Categoría y Expresiones permiten organizar las variables en contextos definidos, lo que facilita su uso en el análisis y evaluación. Este enfoque permite una mayor flexibilidad al asociar preguntas y expresiones a una variable, ofreciendo una estructura de datos más organizada y escalable para las evaluaciones.

```
<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Variable extends Model
{ use HasFactory;
    protected $table = 'variables';
    protected $primaryKey = 'VariableID';
    protected $fillable = ['Nombre', 'CategorialID'];
    public $timestamps = true;

    // Relación con las preguntas
    public function preguntas()
    { return $this->belongsToMany(Pregunta::class, 'pregunta_variable', 'VariableID',
    'PreguntalD'); }

    public function categoria()
    { return $this->belongsTo(Categoria::class, 'CategorialID'); }

    public function expresiones()
    { return $this->hasMany(Expresiones::class, 'VariableID'); } }
```

Controladores

Los controladores en este sistema gestionan las operaciones entre el usuario y la base de datos. Cada uno está diseñado para manejar tareas específicas, como asignar cuestionarios a grupos, almacenar respuestas o calcular puntajes. Por ejemplo, el controlador de Cuestionarios maneja la asignación de cuestionarios, mientras que el de Respuestas Enviadas registra las respuestas de los usuarios. Estos controladores aseguran que los datos sean correctamente procesados y devueltos al usuario.

Controlador usuarios

En este controlador gestionamos varias funcionalidades relacionadas con los usuarios en nuestro sistema. Entre las principales características y funcionalidades que implementamos se encuentran las siguientes:

El método register en el controlador UsuarioController implementa la funcionalidad para registrar usuarios en el sistema. Este proceso inicia verificando los datos enviados por el cliente a través de una solicitud HTTP. Para garantizar que los datos sean correctos y cumplan con las reglas necesarias, se utiliza el validador de Laravel. Aquí se especifican las reglas, como campos obligatorios (Nombre, Matricula, Correo, entre otros), formato de correo válido, unicidad de campos específicos y restricciones para referencias a otras tablas como InstitucionID o CarreraID.


```

<?php
namespace App\Http\Controllers;
use App\Models\Usuario;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Validator;
use App\Helpers\ResponseHelper;

class UsuarioController extends Controller
{ public function register(Request $request)
    { $validator = Validator::make($request->all(), [
        'Nombre' => 'required|string|max:255',
        'Primer_apellido' => 'required|string|max:255',
        'Segundo_apellido' => 'nullable|string|max:255',
        'Matricula' => 'required|string|max:255|unique:usuarios',
        'Semestre' => 'required|string|max:255',
        'Tipo' => 'required|in:Alumno,Maestro,Admin',
        'Correo' => 'required|string|email|max:255|unique:usuarios',
        'Password' => 'required|string|min:8',
        'InstitucionID' => 'nullable|exists:instituciones,InstitucionID',
        'CarreraID' => 'nullable|exists:carreras,CarreraID',
        'edad' => 'nullable|integer|min:1|max:120',
        'genero' => 'nullable|in:Masculino,Femenino,Otro',
        'IDPais' => 'nullable|exists:países,IDPais', ]);

```

Anexo 42 Validación de Datos en el Registro de Usuario

El método register valida los datos enviados en la solicitud. Si hay errores, responde con un código 422 y los detalles de los campos incorrectos. Si los datos son correctos, se crea un nuevo usuario en la base de datos, asegurando la encriptación de la contraseña. Luego, genera un token de autenticación para el usuario, que permite realizar solicitudes autenticadas en el sistema. Finalmente, el método retorna una respuesta con el token y la información del usuario registrado.

```

if ($validator->fails()) { return ResponseHelper::error($validator->errors(), 422); }

// Crear el usuario
$usuario = Usuario::create([
    'Nombre' => $request->Nombre,
    'Primer_apellido' => $request->Primer_apellido,
    'Segundo_apellido' => $request->Segundo_apellido,
    'Matricula' => $request->Matricula,
    'Semestre' => $request->Semestre,
    'Tipo' => $request->Tipo,
    'Correo' => $request->Correo,
    'Password' => Hash::make($request->Password),
    'InstitucionID' => $request->InstitucionID,
    'CarreraID' => $request->CarreraID,
    'edad' => $request->edad,
    'genero' => $request->genero,
    'IDPais' => $request->IDPais, ]);

// Generar token de autenticación
$token = $usuario->createToken('auth_token')->plainTextToken;

return ResponseHelper::success([
    'access_token' => $token,
    'token_type' => 'Bearer',
    'usuario' => $usuario,
], 'Usuario registrado correctamente', 201); }

```

Anexo 43 Generación de Token y Respuesta del Registro

El método show busca un usuario específico por su ID, cargando también las relaciones de carrera, institución y país asociadas a ese usuario. Si el usuario existe, se formatea la respuesta

para incluir no solo los detalles del usuario, sino también la información relacionada, como la institución, la carrera y el país, en forma de objetos. Cada uno de estos detalles es organizado y estructurado de manera que se pueda acceder fácilmente a los atributos relacionados. Finalmente, el método devuelve una respuesta exitosa con todos estos datos, indicando que la operación se realizó correctamente.

```
public function show($id)
{
    $usuario = Usuario::with('carrera', 'institucion', 'pais')->findOrFail($id);

    $formattedUsuario = [
        'UsuarioID' => $usuario->UsuarioID,
        'Nombre' => $usuario->Nombre,
        'Primer_apellido' => $usuario->Primer_apellido,
        'Segundo_apellido' => $usuario->Segundo_apellido,
        'Matricula' => $usuario->Matricula,
        'Semestre' => $usuario->Semestre,
        'Tipo' => $usuario->Tipo,
        'Correo' => $usuario->Correo,
        'edad' => $usuario->edad,
        'genero' => $usuario->genero,
        'InstitucionID' => [
            'InstitucionID' => $usuario->institucion->InstitucionID,
            'Nombre' => $usuario->institucion->Nombre,
            'IDPais' => $usuario->institucion->IDPais,
            'created_at' => $usuario->institucion->created_at,
            'updated_at' => $usuario->institucion->updated_at, ],
        'CarreraID' => [
            'CarreraID' => $usuario->carrera->CarreraID,
            'Nombre' => $usuario->carrera->Nombre,
            'InstitucionID' => $usuario->carrera->InstitucionID,
            'created_at' => $usuario->carrera->created_at,
            'updated_at' => $usuario->carrera->updated_at, ],
    ];
}
```

```
'IDPais' => [ 'IDPais' => $usuario->pais->IDPais,  
              'Nombre' => $usuario->pais->Nombre,  
              'created_at' => $usuario->pais->created_at,  
              'updated_at' => $usuario->pais->updated_at, ],  
'created_at' => $usuario->created_at,  
'updated_at' => $usuario->updated_at,];  
  
return ResponseHelper::success($formattedUsuario, 'Detalles del usuario  
obtenidos correctamente'); }
```

Anexo 45 Respuesta de los detalles de usuario

La función update permite modificar la información de un usuario existente. Primero, busca al usuario con el id proporcionado. Luego, valida los datos que se envían para asegurarse de que cumplen con las reglas definidas, como la unicidad del correo y la matrícula. Si hay errores en los datos, responde con un código 422 y los detalles del error. Si los datos son correctos, actualiza los campos correspondientes del usuario, como nombre, correo, institución y carrera. Si se incluye una nueva contraseña, esta se encripta y se actualiza. Finalmente, devuelve el usuario actualizado con un mensaje de éxito

```

public function update(Request $request, $id)
{
    $usuario = Usuario::findOrFail($id);

    $validator = Validator::make($request->all(), [

        'Nombre' => 'sometimes|string|max:255',
        'Primer_apellido' => 'sometimes|string|max:255',
        'Segundo_apellido' => 'sometimes|nullable|string|max:255',
        'Matricula' => 'sometimes|string|max:255|unique:usuarios,Matricula,' . $id .
        ',UsuarioID',
        'Semestre' => 'sometimes|string|max:255',
        'Tipo' => 'sometimes|in:Alumno,Maestro,Admin',
        'Correo' => 'sometimes|string|email|max:255|unique:usuarios,Correo,' . $id .
        ',UsuarioID',
        'Password' => 'sometimes|string|min:8',
        'edad' => 'sometimes|integer|min:1|max:120',
        'genero' => 'sometimes|in:Masculino,Femenino,Otro',
        'InstitucionID' => 'sometimes|nullable|exists:instituciones,InstitucionID',
        'CarreralID' => 'sometimes|nullable|exists:carreras,CarreralID',
        'IDPais' => 'sometimes|nullable|exists:países,IDPais', ]);

    if ($validator->fails()) { return ResponseHelper::error($validator->errors(), 422); }

    $usuario->update($request->only([

        'Nombre', 'Primer_apellido', 'Segundo_apellido', 'Matricula', 'Semestre', 'Tipo',
        'Correo', 'InstitucionID', 'CarreralID', 'edad', 'genero', 'IDPais']));

    if ($request->has('Password')) {$usuario->update(['Password' =>
    Hash::make($request->Password)]); }

    return ResponseHelper::success($usuario, 'Usuario actualizado correctamente');}

```

La función `usuarios` obtiene todos los usuarios de la base de datos junto con sus relaciones a carrera, institución y país utilizando el método `with`. Luego, la respuesta se formatea para incluir los detalles relevantes de cada usuario, como nombre, matrícula y correo, además de las relaciones correspondientes, como los datos de la institución, carrera y país, si están presentes. Si alguna de estas relaciones es nula, se maneja adecuadamente para evitar errores.

```
public function usuarios()
{
    $usuarios = Usuario::with('carrera', 'institucion', 'pais')->get();

    $formattedUsuarios = $usuarios->map(function ($usuario) {
        return [
            'UsuarioID' => $usuario->UsuarioID,
            'Nombre' => $usuario->Nombre,
            'Primer_apellido' => $usuario->Primer_apellido,
            'Segundo_apellido' => $usuario->Segundo_apellido,
            'Matricula' => $usuario->Matricula,
            'Semestre' => $usuario->Semestre,
            'Tipo' => $usuario->Tipo,
            'Correo' => $usuario->Correo,
            'edad' => $usuario->edad,
            'genero' => $usuario->genero,

            'InstitucionID' => $usuario->institucion ? [
                'InstitucionID' => $usuario->institucion->InstitucionID,
                'Nombre' => $usuario->institucion->Nombre,
                'IDPais' => $usuario->institucion->IDPais,
                'created_at' => $usuario->institucion->created_at,
                'updated_at' => $usuario->institucion->updated_at, ] : null,

            'CarreraID' => $usuario->carrera ? [
                'CarreraID' => $usuario->carrera->CarreraID,
                'Nombre' => $usuario->carrera->Nombre,
                'InstitucionID' => $usuario->carrera->InstitucionID,
                'created_at' => $usuario->carrera->created_at,
                'updated_at' => $usuario->carrera->updated_at, ] : null,
        ];
    });
}
```

Al final, la función devuelve una lista organizada de usuarios, con todos los datos y relaciones necesarias, en formato JSON.

```
'IDPais' => $usuario->pais ? [  
    'IDPais' => $usuario->pais->IDPais,  
    'Nombre' => $usuario->pais->Nombre,  
    'created_at' => $usuario->pais->created_at,  
    'updated_at' => $usuario->pais->updated_at, ] : null,  
  
    'created_at' => $usuario->created_at,  
    'updated_at' => $usuario->updated_at, ]; }  
  
return ResponseHelper::success($formattedUsuarios, 'Lista de usuarios  
obtenida correctamente'); }
```

Anexo 48 Respuesta de la obtención de datos

La función eliminar busca un usuario en la base de datos utilizando su id y, si lo encuentra, lo elimina. Esto se logra a través del método findOrFail, que devuelve el usuario o lanza un error si no se encuentra. Después de eliminar el usuario, se retorna una respuesta con un mensaje de éxito indicando que el usuario fue eliminado correctamente. Esta función es útil para mantener la base de datos actualizada al eliminar registros innecesarios.

```
public function eliminar($id)  
{ $usuario = Usuario::findOrFail($id);  
  $usuario->delete();  
  
  return ResponseHelper::success(null, 'Usuario eliminado correctamente'); }
```

Anexo 49 Eliminación de un usuario

La función obtenerAlumnos se encarga de obtener todos los usuarios de tipo "Alumno", incluyendo sus relaciones con carrera, institución y país. Utilizamos el método with para cargar estas relaciones, y luego filtra los usuarios por tipo "Alumno". Para luego, formatear la respuesta que devuelve los datos estructurados, incluyendo información de cada relación como InstitucionID, CarreraID, y IDPais.

```

public function obtenerAlumnos()
{
    $alumnos = Usuario::with('carrera', 'institucion', 'pais')
        ->where('Tipo', 'Alumno') ->get();

    $formattedAlumnos = $alumnos->map(function ($alumno) {
        return [
            'UsuarioID' => $alumno->UsuarioID,
            'Nombre' => $alumno->Nombre,
            'Primer_apellido' => $alumno->Primer_apellido,
            'Segundo_apellido' => $alumno->Segundo_apellido,
            'Matricula' => $alumno->Matricula,
            'Semestre' => $alumno->Semestre,
            'Tipo' => $alumno->Tipo,
            'Correo' => $alumno->Correo,
            'edad' => $alumno->edad,
            'genero' => $alumno->genero,
            'InstitucionID' => [
                'InstitucionID' => $alumno->institucion->InstitucionID,
                'Nombre' => $alumno->institucion->Nombre,
                'IDPais' => $alumno->institucion->IDPais,
                'created_at' => $alumno->institucion->created_at,
                'updated_at' => $alumno->institucion->updated_at, ],
            'CarreraID' => [
                'CarreraID' => $alumno->carrera->CarreraID,
                'Nombre' => $alumno->carrera->Nombre,
                'InstitucionID' => $alumno->carrera->InstitucionID,
                'created_at' => $alumno->carrera->created_at,
                'updated_at' => $alumno->carrera->updated_at, ],
        ];
    });
}

```


. Finalmente, la función retorna una respuesta con la lista de alumnos, mostrando la información de manera organizada. Esto permite obtener y presentar fácilmente los detalles de todos los alumnos en el sistema.

```
'IDPais' => [  
    'IDPais' => $alumno->pais->IDPais,  
    'Nombre' => $alumno->pais->Nombre,  
    'created_at' => $alumno->pais->created_at,  
    'updated_at' => $alumno->pais->updated_at, ],  
    'created_at' => $alumno->created_at,  
    'updated_at' => $alumno->updated_at, ]; });  
  
return ResponseHelper::success($formattedAlumnos, 'Lista de alumnos obtenida  
correctamente'); }
```

Anexo 51 Respuesta de la búsqueda de alumnos

La función *obtenerMaestros* obtiene todos los usuarios de tipo "Maestro" junto con sus relaciones a carrera, institución y país. Primero, utiliza el método *with* para cargar estas relaciones. Luego, filtra los usuarios por el tipo "Maestro". Tras obtener los datos, la función formatea la respuesta para devolver la información organizada, incluyendo los detalles de las relaciones, como el *InstitucionID*, *CarreraID* y *IDPais*, si existen.

```
public function obtenerMaestros()  
{ $maestros = Usuario::with('carrera', 'institucion', 'pais') ->where('Tipo', 'Maestro') ->get();  
    $formattedMaestros = $maestros->map(function ($maestro) {  
        return [ 'Usuarioid' => $maestro->Usuarioid,  
            'Nombre' => $maestro->Nombre,  
            'Primer_apellido' => $maestro->Primer_apellido,  
            'Segundo_apellido' => $maestro->Segundo_apellido,  
            'Matricula' => $maestro->Matricula,  
            'Semestre' => $maestro->Semestre,  
            'Tipo' => $maestro->Tipo,  
            'Correo' => $maestro->Correo,  
            'edad' => $maestro->edad,  
            'genero' => $maestro->genero,
```

Anexo 52 Obtener maestros

Finalmente, la función retorna una respuesta con la lista de maestros con todos los detalles relevantes, lo que facilita la visualización de la información de manera estructurada.

```
'InstitucionID' => $maestro->institucion ? [  
    'InstitucionID' => $maestro->institucion->InstitucionID,  
    'Nombre' => $maestro->institucion->Nombre,  
    'IDPais' => $maestro->institucion->IDPais,  
    'created_at' => $maestro->institucion->created_at,  
    'updated_at' => $maestro->institucion->updated_at, ] : null,  
  
    'CarreraID' => $maestro->carrera ? [  
        'CarreraID' => $maestro->carrera->CarreraID,  
        'Nombre' => $maestro->carrera->Nombre,  
        'InstitucionID' => $maestro->carrera->InstitucionID,  
        'created_at' => $maestro->carrera->created_at,  
        'updated_at' => $maestro->carrera->updated_at, ] : null,  
  
    'IDPais' => $maestro->pais ? [  
        'IDPais' => $maestro->pais->IDPais,  
        'Nombre' => $maestro->pais->Nombre,  
        'created_at' => $maestro->pais->created_at,  
        'updated_at' => $maestro->pais->updated_at, ] : null,  
    'created_at' => $maestro->created_at,  
    'updated_at' => $maestro->updated_at, ]; }  
  
return ResponseHelper::success($formattedMaestros, 'Lista de maestros  
obtenida correctamente'); }
```

Anexo 53 Respuesta de la obtención de información del maestro

La función `obtenerAdministradores` obtiene todos los usuarios con el tipo "Admin" y sus relaciones correspondientes a carrera, institución y país. Utiliza `with` para cargar las relaciones necesarias

y filtra los usuarios por tipo "Admin". Luego, mapea los resultados obtenidos, formateando la respuesta para incluir los detalles de cada administrador, como su nombre, matrícula, correo y otras propiedades, así como los datos relacionados con su institución, carrera y país, si están disponibles.

```
public function obtenerAdministradores()
{
    $administradores = Usuario::with('carrera', 'institucion', 'pais')
        ->where('Tipo', 'Admin') ->get();

    $formattedAdministradores = $administradores->map(function ($admin) {
        return [ 'UsuarioID' => $admin->UsuarioID,
            'Nombre' => $admin->Nombre,
            'Primer_apellido' => $admin->Primer_apellido,
            'Segundo_apellido' => $admin->Segundo_apellido,
            'Matricula' => $admin->Matricula,
            'Semestre' => $admin->Semestre,
            'Tipo' => $admin->Tipo,
            'Correo' => $admin->Correo,
            'edad' => $admin->edad,
            'genero' => $admin->genero,

            'InstitucionID' => $admin->institucion ? [
                'InstitucionID' => $admin->institucion->InstitucionID,
                'Nombre' => $admin->institucion->Nombre,
                'IDPais' => $admin->institucion->IDPais,
                'created_at' => $admin->institucion->created_at,
                'updated_at' => $admin->institucion->updated_at, ] : null,
        ];
    });
}
```

Anexo 54 Obtención de datos de los administradores

Finalmente, la función retorna una respuesta exitosa con la lista de administradores formateada correctamente.

```

'CarreraID' => $admin->carrera ? [
    'CarreraID' => $admin->carrera->CarreraID,
    'Nombre' => $admin->carrera->Nombre,
    'InstitucionID' => $admin->carrera->InstitucionID,
    'created_at' => $admin->carrera->created_at,
    'updated_at' => $admin->carrera->updated_at, ] : null,

'IDPais' => $admin->pais ? [
    'IDPais' => $admin->pais->IDPais,
    'Nombre' => $admin->pais->Nombre,
    'created_at' => $admin->pais->created_at,
    'updated_at' => $admin->pais->updated_at, ] : null,
'created_at' => $admin->created_at,
'updated_at' => $admin->updated_at, ]; });

return ResponseHelper::success($formattedAdministradores, 'Lista de
Administradores obtenida correctamente'); }

```

Anexo 55 Respuesta de los datos del administrador

La función buscarMaestroPorInstitucion permite buscar maestros que pertenecen a una institución específica. Filtra los usuarios con tipo "Maestro" y el InstitucionID proporcionado. Si no se encuentran maestros, devuelve un error indicando que no se encontraron resultados. Si existen maestros, se formatea la respuesta para incluir los datos de cada maestro, como su nombre, matrícula, correo y las relaciones de institución, carrera y país, si están presentes.

```

public function buscarMaestroPorInstitucion($institucionId)
{
    $maestros = Usuario::with('carrera', 'institucion', 'pais')
        ->where('Tipo', 'Maestro') ->where('InstitucionID', $institucionId) ->get();

    if ($maestros->isEmpty()) {
        return ResponseHelper::error('No se encontraron maestros en esta institución',
            404); }

    $formattedMaestros = $maestros->map(function ($maestro) {
        return [ 'UsuarioID' => $maestro->UsuarioID,
            'Nombre' => $maestro->Nombre,
            'Primer_apellido' => $maestro->Primer_apellido,
            'Segundo_apellido' => $maestro->Segundo_apellido,
            'Matricula' => $maestro->Matricula,
            'Semestre' => $maestro->Semestre,
            'Tipo' => $maestro->Tipo,
            'Correo' => $maestro->Correo,
            'edad' => $maestro->edad,
            'genero' => $maestro->genero,

            'InstitucionID' => $maestro->institucion ? [
                'InstitucionID' => $maestro->institucion->InstitucionID,
                'Nombre' => $maestro->institucion->Nombre,
                'IDPais' => $maestro->institucion->IDPais,
                'created_at' => $maestro->institucion->created_at,
                'updated_at' => $maestro->institucion->updated_at, ] : null,
        ];
    });
}

```

Anexo 56 Búsqueda de los datos del maestro por institución

Finalmente, la función devuelve una respuesta exitosa con la lista de maestros obtenida y formateada correctamente.

```

'CarreraID' => $maestro->carrera ? [
    'CarreraID' => $maestro->carrera->CarreraID,
    'Nombre' => $maestro->carrera->Nombre,
    'InstitucionID' => $maestro->carrera->InstitucionID,
    'created_at' => $maestro->carrera->created_at,
    'updated_at' => $maestro->carrera->updated_at, ] : null,

'IDPais' => $maestro->pais ? [
    'IDPais' => $maestro->pais->IDPais,
    'Nombre' => $maestro->pais->Nombre,
    'created_at' => $maestro->pais->created_at,
    'updated_at' => $maestro->pais->updated_at, ] : null,

'created_at' => $maestro->created_at,
'updated_at' => $maestro->updated_at, ]; });

return ResponseHelper::success($formattedMaestros, 'Lista de maestros obtenida
correctamente'); }

```

Anexo 57 Respuesta de la búsqueda

La función buscarAlumnoPorInstitucion permite buscar alumnos por el InstitucionID proporcionado. Filtra los usuarios con tipo "Alumno" y el InstitucionID específico. Si no se encuentran alumnos, devuelve un error con un mensaje adecuado. En caso de encontrar resultados, se formatean los datos de los alumnos, incluyendo información básica como nombre, matrícula, semestre y datos de las relaciones de institución, carrera y país, si están disponibles.

```

public function buscarAlumnoPorInstitucion($institucionId)
{
    $alumnos = Usuario::with('carrera', 'institucion', 'pais')
        ->where('Tipo', 'Alumno') ->where('InstitucionID', $institucionId) ->get();

    if ($alumnos->isEmpty()) {
        return ResponseHelper::error('No se encontraron alumnos en esta institución',
            404); }

    $formattedAlumnos = $alumnos->map(function ($alumno) {
        return [ 'UsuarioID' => $alumno->UsuarioID,
            'Nombre' => $alumno->Nombre,
            'Primer_apellido' => $alumno->Primer_apellido,
            'Segundo_apellido' => $alumno->Segundo_apellido,
            'Matricula' => $alumno->Matricula,
            'Semestre' => $alumno->Semestre,
            'Tipo' => $alumno->Tipo,
            'Correo' => $alumno->Correo,
            'edad' => $alumno->edad,
            'genero' => $alumno->genero,

            'InstitucionID' => $alumno->institucion ? [
                'InstitucionID' => $alumno->institucion->InstitucionID,
                'Nombre' => $alumno->institucion->Nombre,
                'IDPais' => $alumno->institucion->IDPais,
                'created_at' => $alumno->institucion->created_at,
                'updated_at' => $alumno->institucion->updated_at, ] : null,
        ];
    });
}

```

Anexo 58 Búsqueda de alumnos por institución

Finalmente, la función retorna una respuesta exitosa con la lista de alumnos formateada correctamente.

```

'CarreraID' => $alumno->carrera ? [
    'CarreraID' => $alumno->carrera->CarreraID,
    'Nombre' => $alumno->carrera->Nombre,
    'InstitucionID' => $alumno->carrera->InstitucionID,
    'created_at' => $alumno->carrera->created_at,
    'updated_at' => $alumno->carrera->updated_at, ] : null,

'IDPais' => $alumno->pais ? [
    'IDPais' => $alumno->pais->IDPais,
    'Nombre' => $alumno->pais->Nombre,
    'created_at' => $alumno->pais->created_at,
    'updated_at' => $alumno->pais->updated_at, ] : null,
'created_at' => $alumno->created_at,
'updated_at' => $alumno->updated_at, ]; });

return ResponseHelper::success($formattedAlumnos, 'Lista de alumnos obtenida
correctamente'); }

```

Anexo 59 Respuesta de la búsqueda de alumnos

La función login valida los datos de inicio de sesión del usuario, asegurándose de que se proporcionen el correo electrónico o matrícula y la contraseña. Luego, busca al usuario en la base de datos utilizando el correo o matrícula proporcionada. Si el usuario no existe o la contraseña no coincide, se devuelve un error con un mensaje de "Credenciales incorrectas". Si la autenticación es exitosa, se genera un token de acceso para el usuario utilizando el método createToken, que permite la autenticación en futuras solicitudes. Finalmente, se retorna una respuesta JSON con el token y los datos del usuario autenticado.


```

public function login(Request $request)
{
    $request->validate([ 'Correo' => 'required|string', 'Password' => 'required|string', ]);

    $usuario = Usuario::where('Correo', $request->Correo)
        ->orWhere('Matricula', $request->Correo)
        ->first();

    if (!$usuario || !Hash::check($request->Password, $usuario->Password)) {
        return response()->json(['message' => 'Credenciales incorrectas'], 401); }

    // Crear un token para el usuario autenticado
    $token = $usuario->createToken('auth_token')->plainTextToken;
    return response()->json([ 'access_token' => $token, 'token_type' => 'Bearer',
        'usuario' => $usuario, ]); }

```

Anexo 60 Login de usuarios

Controlador países

El método agregarPais valida los datos de entrada para asegurarse de que el nombre del país sea único y cumpla con las restricciones de longitud. Si la validación falla, se devuelve un mensaje de error con el código 400. Si la validación es exitosa, crea una nueva instancia del modelo Pais y guarda el nombre del país en la base de datos. Si el país se guarda correctamente, se devuelve una respuesta de éxito con el país creado y el código 201. Si ocurre un error al guardar, se devuelve un mensaje de error con el código 500.

```

<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Helpers\ResponseHelper;
use App\Models\Paises;
use Illuminate\Support\Facades\Validator;
class PaisesController extends Controller
{ public function agregarPais(Request $request)
    { $validator = Validator::make($request->all(), [
        'Nombre' => 'required|string|max:255|unique:paises,Nombre', ]);

    if ($validator->fails()) {
        return ResponseHelper::error($validator->errors()->first(), 400); }

    $pais = new Paises();
    $pais->Nombre = $request->input('Nombre');

    if ($pais->save()) {
        return ResponseHelper::success($pais, 'País creado exitosamente', 201);
    } else {
        return ResponseHelper::error('Error al crear el país', 500); } }

```

Anexo 61 Controlador para crear países

El método actualizarPais valida los datos de entrada para asegurar que el nombre del país sea único, excepto para el país que se está actualizando. Si la validación falla, devuelve un error con el código 400. Luego, busca el país por su ID, y si no lo encuentra, devuelve un error con el código 404. Si encuentra el país, actualiza el nombre y guarda los cambios en la base de datos. Si la actualización es exitosa, se devuelve una respuesta de éxito, de lo contrario, se devuelve un error con el código 500.

```

public function actualizarPais(Request $request, $id)
{
    $validator = Validator::make($request->all(), [
        'Nombre' => 'required|string|max:255|unique:países,Nombre,' . $id . ',IDPaís', ]);

    if ($validator->fails()) {
        return ResponseHelper::error($validator->errors()->first(), 400); }

    $pais = Países::find($id);

    if (!$pais) {
        return ResponseHelper::error('País no encontrado', 404); }

    $pais->Nombre = $request->input('Nombre');

    if ($pais->save()) {
        return ResponseHelper::success($pais, 'País actualizado correctamente');
    } else {
        return ResponseHelper::error('Error al actualizar el país', 500); } }

```

Anexo 62 Actualización del país

El método países obtiene todos los países registrados en la base de datos utilizando el modelo Países::all(). Luego, devuelve una respuesta exitosa que contiene la lista completa de países. La respuesta se estructura con un mensaje que indica que los países se obtuvieron correctamente. Este método es útil cuando se necesita acceder a todos los registros de países almacenados. Si se realiza correctamente, el sistema devuelve los datos en formato adecuado para el cliente.

```

public function países()
{
    $países = Países::all();

    return ResponseHelper::success($países, 'Países obtenidos correctamente'); }

```

Anexo 63 Obtención de todos los países

El método país busca un país específico en la base de datos utilizando el ID proporcionado. Si no se encuentra el país, retorna un error con un mensaje de "País no encontrado" y un código de estado 404. Si el país existe, se devuelve una respuesta exitosa con los datos del país solicitado. Este método permite obtener información detallada de un país a partir de su identificador único. Es útil cuando se necesita acceder a datos específicos y gestionar casos donde no se encuentra el país.

```
public function pais($id)
{
    $paises = Paises::find($id);

    if (!$paises) { return ResponseHelper::error('Pais no encontrado', 404); }

    return ResponseHelper::success($paises, 'Pais obtenido correctamente'); }
```

Anexo 64 Obtención de una institución específica

El método eliminarPais busca un país en la base de datos utilizando el ID proporcionado. Si el país existe, se elimina de la base de datos utilizando el método delete(). Si no se encuentra el país, se lanza una excepción findOrFail. Al finalizar la operación, se devuelve una respuesta exitosa con un mensaje indicando que el país fue eliminado correctamente

```
public function eliminarPais($id)
{
    $pais = Paises::findOrFail($id); $pais->delete();

    return ResponseHelper::success(null, 'Pais eliminado correctamente'); }
```

Anexo 65 Eliminación país

El método buscarPaises valida que se haya proporcionado un parámetro de búsqueda, que es un string. Si la validación falla, se devuelve un error con el mensaje correspondiente. Luego, se obtiene el valor del parámetro query y se busca en la base de datos los países cuyo nombre coincida parcialmente con el término proporcionado, utilizando el operador LIKE. Si no se encuentran países que coincidan, se devuelve un error. Finalmente, si se encuentran resultados, se devuelve una respuesta exitosa con los países que coinciden con la búsqueda.

```

public function buscarPaises(Request $request)
{
    $validator = Validator::make($request->all(), [
        'query' => 'required|string|max:255',
    ]);

    if ($validator->fails()) {
        return ResponseHelper::error($validator->errors()->first(), 400); }

    $query = $request->input('query');

    $paises = Paises::where('nombre', 'LIKE', '%' . $query . '%')->get();

    if ($paises->isEmpty()) {
        return ResponseHelper::error('No se encontraron países que coincidan con la búsqueda', 404); }

    return ResponseHelper::success($paises, 'Resultados de la búsqueda obtenidos correctamente'); } }

```

Anexo 66 Búsqueda de países

Controlador instituciones

El método crearInstitucion valida los datos de entrada y crea una nueva institución asociada a un país. En el caso de que la validación falle, devuelve un error, y si la creación es exitosa, se guarda y retorna la institución creada.

```

<?php
namespace App\Http\Controllers;
use App\Models\Instituciones;
use Illuminate\Support\Facades\Validator;
use Illuminate\Http\Request;
use App\Helpers\ResponseHelper;

class InstitucionesController extends Controller
{ public function crearInstitucion(Request $request)
    { $validator = Validator::make($request->all(), [
        'Nombre' => 'required|string|max:255',
        'IDPais' => 'required|integer|exists:paises,IDPais', ]);

        if ($validator->fails()) {
            return ResponseHelper::error($validator->errors()->first(), 400); }

        $institucion = new Instituciones();
        $institucion->Nombre = $request->input('Nombre');
        $institucion->IDPais = $request->input('IDPais');

        if ($institucion->save()) {
            return ResponseHelper::success([
                'institucion' => $institucion
            ], 'Institución creada exitosamente', 201);
        } else {
            return ResponseHelper::error('Error al crear la institución', 500); } }

```

Anexo 67 Método para la creación de instituciones

Luego, el método instituciones obtiene todas las instituciones, mientras que institucion busca una institución específica por su ID, retornando un error si no se encuentra.

```

// Método para obtener todas las instituciones
public function instituciones()
{
    $instituciones = Instituciones::all();
    return ResponseHelper::success($instituciones, 'Instituciones obtenidas correctamente');
}

// Método para obtener las instuciones por ID
public function institucion($id)
{
    $institucion = Instituciones::find($id);
    if (!$institucion) {
        return ResponseHelper::error('Institución no encontrada', 404);}

    return ResponseHelper::success([
        'institucion' => $institucion
    ], 'Institución obtenida correctamente');
}

```

Anexo 68 Método para la búsqueda de instituciones

También se incluyen métodos como institucionesPorPais, que filtra las instituciones por país,

```

public function instituciones()
{
    $instituciones = Instituciones::all();
    return ResponseHelper::success($instituciones, 'Instituciones obtenidas correctamente');
}

```

```

6 // Método para obtener instituciones filtradas por país
7 1 reference | 0 overrides
8 public function institucionesPorPaís(Request $request): JsonResponse|mixed
9 {
10     $IDPaís = $request->query(key: 'IDPaís');
11
12     // Validar si se proporciona el IDPaís
13     if (!$IDPaís) {
14         return ResponseHelper::error(message: 'IDPaís es requerido', status: 400);
15     }
16
17     // Filtrar instituciones por el ID del país
18     $instituciones = Instituciones::where(column: 'IDPaís', operator: $IDPaís)->get();
19
20     // Devolver una respuesta exitosa con las instituciones filtradas
21     return ResponseHelper::success(data: $instituciones, message: 'Instituciones obtenidas correctamente');
22 }

```

Anexo 69 Método de búsqueda de instituciones por país

El método *actualizarInstitucion*, que actualiza el nombre de una institución, manejando también posibles errores.

```

1 reference | 0 overrides
2 public function actualizarInstitucion(Request $request, $id): JsonResponse|mixed
3 {
4     // Validación de los datos entrantes
5     $validator = Validator::make($request->all(), [
6         'Nombre' => 'required|string|max:255',
7     ]);
8
9     // Si la validación falla, devolver errores
10    if ($validator->fails()) {
11        return ResponseHelper::error(message: $validator->errors()->first(), status: 400);
12    }
13
14    // Buscar la institución por el campo InstitucionID en lugar de 'id'
15    $institucion = Instituciones::find(id: $id);
16
17    if (!$institucion) {
18        return ResponseHelper::error(message: 'Institución no encontrada', status: 404);
19    }
20
21    // Actualizar el nombre de la institución
22    $institucion->Nombre = $request->input(key: 'Nombre');
23
24    // Guardar los cambios en la base de datos
25    if ($institucion->save()) {
26        return ResponseHelper::success(data: [
27            'institucion' => $institucion
28        ], message: 'Nombre de la institución actualizado correctamente');
29    } else {
30        return ResponseHelper::error(message: 'Error al actualizar el nombre de la institución', status: 500);
31    }
32 }

```

Anexo 13 Método para actualizar instituciones

Finalmente, el método *destroy* elimina una institución por su ID, asegurando la correcta eliminación.


```

// Método para eliminar una institución
1 reference | 0 overrides
public function destroy($id): JsonResponse|mixed
{
    $institucion = Instituciones::findOrFail(id: $id);
    $institucion->delete();

    return ResponseHelper::success(data: null, message: 'Institución eliminada correctamente');
}
}

```

Anexo 14 Método para eliminar instituciones

Controlador carreras

En este código se implementan varios métodos relacionados con la gestión de carreras en una aplicación. El método `agregarCarrera` valida los datos proporcionados y crea una nueva carrera asociada a una institución, devolviendo una respuesta exitosa. El método `editarCarrera` permite modificar una carrera existente, primero validando los datos y luego actualizando la carrera en la base de datos.

```

BackendCEP > app > Http > Controllers > CarrerasController.php > ...
3 namespace App\Http\Controllers;
4
5 use App\Models\Carreras;
6 use Illuminate\Http\Request;
7 use App\Helpers\ResponseHelper;
8
9 7 references | 0 implementations
10 class CarrerasController extends Controller
11 {
12     // Método para agregar una nueva carrera
13     1 reference | 0 overrides
14     public function agregarCarrera(Request $request): JsonResponse|mixed
15     {
16         $validatedData = $request->validate(rules: [
17             'Nombre' => 'required|string|max:255',
18             'InstitucionID' => 'required|exists:instituciones,InstitucionID',
19         ]);
20         $carrera = Carreras::create(attributes: $validatedData);
21         return ResponseHelper::success(data: $carrera, message: 'Carrera creada correctamente', status: 201);
22     }
23
24     // Método para editar una carrera existente
25     1 reference | 0 overrides
26     public function editarCarrera(Request $request, $id): JsonResponse|mixed
27     {
28         $carrera = Carreras::find(id: $id);
29         if (!$carrera) {
30             return ResponseHelper::error(message: 'Carrera no encontrada', status: 404);
31         }
32         $validatedData = $request->validate(rules: [
33             'Nombre' => 'sometimes|required|string|max:255',
34             'InstitucionID' => 'sometimes|required|exists:instituciones,InstitucionID',
35         ]);
36         $carrera->update($validatedData);
37         return ResponseHelper::success(data: $carrera, message: 'Carrera actualizada correctamente');
38     }
39 }

```

Anexo 15 Método para agregar y editar una carrera

Por otro lado, `carreras` obtiene todas las carreras, mientras que `carrera` busca una carrera específica por su ID y maneja el error si no se encuentra.

```

BackendCEP > app > Http > Controllers > CarrerasController.php > ...
9 class CarrerasController extends Controller
24 public function editarCarrera(Request $request, $id): JsonResponse|mixed
36 }
37
38 // Método para obtener todas las carreras
39 1 reference | 0 overrides
40 public function carreras(): JsonResponse|mixed
41 {
42     $carreras = Carreras::all();
43
44     // Devolver una respuesta exitosa con los datos
45     return ResponseHelper::success(data: $carreras, message: 'Carreras obtenidas correctamente');
46 }
47
48 // Método para obtener una carrera específica por ID
49 1 reference | 0 overrides
50 public function carrera($id): JsonResponse|mixed
51 {
52     $carrera = Carreras::find(id: $id);
53
54     if (!$carrera) {
55         // Devolver un error si no se encuentra la carrera
56         return ResponseHelper::error(message: 'Carrera no encontrada', status: 404);
57     }
58
59     // Devolver una respuesta exitosa con la carrera
60     return ResponseHelper::success(data: $carrera, message: 'Carrera obtenida correctamente');
61 }

```

Anexo 16 Métodos de búsqueda para las carreras

El método carrerasPorInstitucion filtra las carreras según el ID de la institución proporcionado, y finalmente, eliminar elimina una carrera existente por su ID.

```

60
61 // Método para obtener carreras filtradas por institución
62 1 reference | 0 overrides
63 public function carrerasPorInstitucion(Request $request): JsonResponse|mixed
64 {
65     $InstitucionID = $request->query(key: 'InstitucionID');
66
67     // Validar si se proporciona el InstitucionID
68     if (!$InstitucionID) {
69         return ResponseHelper::error(message: 'InstitucionID es requerido', status: 400);
70     }
71
72     // Filtrar carreras por el ID de la institución
73     $carreras = Carreras::where(column: 'InstitucionID', operator: $InstitucionID->get());
74
75     // Devolver una respuesta exitosa con las carreras filtradas
76     return ResponseHelper::success(data: $carreras, message: 'Carreras obtenidas correctamente');
77 }
78
79 // Eliminar una carrera
80 1 reference | 0 overrides
81 public function eliminar($id): JsonResponse|mixed
82 {
83     $carrera = Carreras::findOrFail(id: $id);
84     $carrera->delete();
85     return ResponseHelper::success(data: null, message: 'Carrera eliminada correctamente');
86 }
87 }

```

Anexo 17 Método para filtrar carreras por instituciones y eliminarlas

Controlador cuestionario

El método getPreguntasRespuestas primero busca el cuestionario especificado por su ID, cargando sus grupos, preguntas y las respuestas asociadas a cada una. Si no se encuentra el cuestionario, se retorna un error 404. Luego, el código toma las primeras 55 preguntas del cuestionario y estructura la respuesta para incluir detalles como el nombre, descripción, fechas de publicación y cierre de

los grupos, y las preguntas con sus respectivas respuestas y variables. Esto se organiza en un formato JSON para su consumo posterior.

```
BackendCEP > app > Http > Controllers > CuestionarioController.php > ...
5 use Illuminate\Http\Request;
6 use App\Models\Pregunta;
7 use App\Models\CuestionarioPregunta;
8 use App\Models\Cuestionario;
9 use App\Models\Grupos;
10 use App\Models\Respuesta;
11 use App\Models\CuestionarioGrupo;
12
13 10 references | 0 implementations
14 class CuestionarioController extends Controller
15 {
16     1 reference | 0 overrides
17     public function getPreguntasRespuestas($cuestionarioID): JsonResponse|mixed
18     {
19         // Metodo para encontrar el cuestionario y sus preguntas, incluyendo las respuestas y las variables.
20         $cuestionario = Cuestionario::with(relations: ['grupo', 'preguntas.respuestas', 'preguntas.variables'])
21             ->where(column: 'CuestionarioID', operator: $cuestionarioID)
22             ->first();
23
24         if (!$cuestionario) {
25             return response()->json(['message' => 'Cuestionario no encontrado'], 404);
26         }
27
28         // Tomar solo las primeras 55 preguntas
29         $preguntas = $cuestionario->preguntas->take(55);
30
31         // Estructura de la respuesta para incluir el nombre, la descripción, las preguntas, las respuestas y las variables.
32         return response()->json([
33             'CuestionarioID' => $cuestionario->CuestionarioID,
34             'nombre' => $cuestionario->Nombre,
35             'descripcion' => $cuestionario->Descripcion,
36             'fechas' => $cuestionario->cuestionarioGrupos->map(function ($grupo): array {
37                 return [
38                     'GrupoNombre' => $grupo->grupo->Nombre ?? 'Sin grupo',
39                     'FechaPublicacion' => $grupo->FechaPublicacion,
40                     'FechaCierre' => $grupo->FechaCierre,
41                 ];
42             });
43         ]);
44     }
45 }
```

Anexo 18 Método para encontrar un cuestionario con sus preguntas

```
BackendCEP > app > Http > Controllers > CuestionarioController.php > ...
13 class CuestionarioController extends Controller
14 {
15     public function getPreguntasRespuestas($cuestionarioID): JsonResponse|mixed
16     {
17         // Metodo para encontrar el cuestionario y sus preguntas, incluyendo las respuestas y las variables.
18         $cuestionario = Cuestionario::with(relations: ['grupo', 'preguntas.respuestas', 'preguntas.variables'])
19             ->where(column: 'CuestionarioID', operator: $cuestionarioID)
20             ->first();
21
22         if (!$cuestionario) {
23             return response()->json(['message' => 'Cuestionario no encontrado'], 404);
24         }
25
26         // Tomar solo las primeras 55 preguntas
27         $preguntas = $cuestionario->preguntas->take(55);
28
29         // Estructura de la respuesta para incluir el nombre, la descripción, las preguntas, las respuestas y las variables.
30         return response()->json([
31             'CuestionarioID' => $cuestionario->CuestionarioID,
32             'nombre' => $cuestionario->Nombre,
33             'descripcion' => $cuestionario->Descripcion,
34             'fechas' => $cuestionario->cuestionarioGrupos->map(function ($grupo): array {
35                 return [
36                     'GrupoNombre' => $grupo->grupo->Nombre ?? 'Sin grupo',
37                     'FechaPublicacion' => $grupo->FechaPublicacion,
38                     'FechaCierre' => $grupo->FechaCierre,
39                 ];
40             });
41             'preguntas' => $preguntas->map(function ($pregunta): array {
42                 return [
43                     'PreguntaID' => $pregunta->PreguntaID,
44                     'Texto' => $pregunta->Texto,
45                     'Tipo' => $pregunta->Tipo,
46                     'variables' => $pregunta->variables->map(function ($variable): array {
47                         return [
48                             'VariableID' => $variable->VariableID,
49                             'Nombre' => $variable->Nombre,
50                         ];
51                     });
52                 ],
53             'respuestas' => $pregunta->respuestas->map(function ($respuesta): array {
54                 return [
55                     'RespuestaID' => $respuesta->RespuestaID,
56                     'Texto' => $respuesta->Texto,
57                     'Valor' => $respuesta->Valor,
58                 ];
59             });
60         ],
61         );
62     }
63 }
```

Anexo 19 Mapeo de las preguntas con sus respuestas

Este código crea un cuestionario a partir de datos proporcionados por el usuario. Primero, valida la entrada para asegurarse de que los datos, como el nombre, descripción y preguntas, sean

correctos. Luego, crea el cuestionario y limita las preguntas a un máximo de 55. A continuación, cada pregunta se asocia con variables y respuestas previamente existentes, utilizando relaciones en las tablas de la base de datos. Finalmente, el cuestionario se guarda y se devuelve un mensaje de éxito junto con los detalles del cuestionario creado.

```
BackendCEP > app > Http > Controllers > CuestionarioController.php > ...
13 class CuestionarioController extends Controller
14 {
15     1 reference | 0 overrides
16     public function crearCuestionario(Request $request): JsonResponse|mixed
17     {
18         // Validar datos de entrada
19         $validatedData = $request->validate(rules: [
20             'nombre' => 'required|string|max:255',
21             'descripcion' => 'nullable|string',
22             'preguntas' => 'required|array',
23             'preguntas.*.Texto' => 'required|string',
24             'preguntas.*.Tipo' => 'required|string|in:Positiva,Negativa',
25             'preguntas.*.variables' => 'required|array',
26             'preguntas.*.variables.*' => 'exists:variables,VariableID',
27         ]);
28
29         // Limitar las preguntas a las primeras 55
30         $preguntas = array_slice(array: $validatedData['preguntas'], offset: 0, length: 55);
31
32         // Crear el cuestionario
33         $cuestionario = Cuestionario::create(attributes: [
34             'Nombre' => $validatedData['nombre'],
35             'Descripcion' => $validatedData['descripcion'],
36         ]);
37
38         // Procesar preguntas y asociarlas al cuestionario
39         foreach ($preguntas as $preguntaData) {
40             // Crear una nueva pregunta
41             $nuevaPregunta = Pregunta::create(attributes: [
42                 'Texto' => $preguntaData['Texto'],
43                 'Tipo' => $preguntaData['Tipo'],
44             ]);
45
46             // Asociar las variables a la nueva pregunta
47             $nuevaPregunta->variables()->sync($preguntaData['variables']);
48
49             // Asociar las respuestas existentes a la nueva pregunta
50             $respuestasExistentes = Respuesta::where(column: 'PreguntaID', operator: 1)->get(); // ID de la pregunta de referencia
51         }
52     }
53 }
```

Anexo 20 Método para crear un cuestionario

```
13 class CuestionarioController extends Controller
14 {
15     public function crearCuestionario(Request $request): JsonResponse|mixed
16     {
17         // Validar datos de entrada
18         $validatedData = $request->validate(rules: [
19             'nombre' => 'required|string|max:255',
20             'descripcion' => 'nullable|string',
21             'preguntas' => 'required|array',
22             'preguntas.*.Texto' => 'required|string',
23             'preguntas.*.Tipo' => 'required|string|in:Positiva,Negativa',
24             'preguntas.*.variables' => 'required|array',
25             'preguntas.*.variables.*' => 'exists:variables,VariableID',
26         ]);
27
28         // Limitar las preguntas a las primeras 55
29         $preguntas = array_slice(array: $validatedData['preguntas'], offset: 0, length: 55);
30
31         // Crear el cuestionario
32         $cuestionario = Cuestionario::create(attributes: [
33             'Nombre' => $validatedData['nombre'],
34             'Descripcion' => $validatedData['descripcion'],
35         ]);
36
37         // Procesar preguntas y asociarlas al cuestionario
38         foreach ($preguntas as $preguntaData) {
39             // Crear una nueva pregunta
40             $nuevaPregunta = Pregunta::create(attributes: [
41                 'Texto' => $preguntaData['Texto'],
42                 'Tipo' => $preguntaData['Tipo'],
43             ]);
44
45             // Asociar las variables a la nueva pregunta
46             $nuevaPregunta->variables()->sync($preguntaData['variables']);
47
48             // Asociar las respuestas existentes a la nueva pregunta
49             $respuestasExistentes = Respuesta::where(column: 'PreguntaID', operator: 1)->get(); // ID de la pregunta de referencia
50             foreach ($respuestasExistentes as $respuesta) {
51                 $respuesta->create(attributes: [
52                     'Texto' => $respuesta->Texto,
53                     'Valor' => $respuesta->Valor,
54                     'PreguntaID' => $nuevaPregunta->PreguntaID,
55                 ]);
56             }
57
58             // Asociar la nueva pregunta al cuestionario
59             CuestionarioPregunta::create(attributes: [
60                 'CuestionarioID' => $cuestionario->CuestionarioID,
61                 'PreguntaID' => $nuevaPregunta->PreguntaID,
62             ]);
63         }
64
65         return response()->json([
66             'message' => 'Cuestionario creado exitosamente',
67             'cuestionario' => $cuestionario,
68         ], 201);
69     }
70 }
```

Anexo 21 Asociación de las preguntas con un cuestionario

Este código obtenemos las versiones de los cuestionarios, incluyendo sus preguntas y respuestas, así como los grupos asociados. Utiliza la relación entre los cuestionarios, sus preguntas y los grupos para estructurar la respuesta. Cada cuestionario incluye detalles como su nombre, descripción y fechas de publicación y cierre de los grupos. Finalmente, devuelve esta información en un formato JSON que organiza todos estos datos de manera estructurada.

```
BackendCEP > app > Http > Controllers > CuestionarioController.php > ...
13 class CuestionarioController extends Controller
    2 references | 0 overrides
125 public function getQuizVersions(): JsonResponse|mixed
126 {
127     $cuestionarios = Cuestionario::with(relations: ['preguntas.respuestas', 'cuestionarioGrupos.grupo'])
128     ->get();
129
130     return response()->json([
131         'cuestionarios' => $cuestionarios->map(callback: function (Cuestionario $cuestionario): array {
132             return [
133                 'CuestionarioID' => $cuestionario->CuestionarioID,
134                 'nombre' => $cuestionario->Nombre,
135                 'descripcion' => $cuestionario->Descripcion,
136                 'fechas' => $cuestionario->cuestionarioGrupos->map(function ($grupo): array {
137                     return [
138                         'GrupoNombre' => $grupo->grupo->Nombre ?? 'Sin grupo', // Si el grupo no existe
139                         'FechaPublicacion' => $grupo->FechaPublicacion,
140                         'FechaCierre' => $grupo->FechaCierre,
141                     ];
142                 })),
143                 'preguntas' => $cuestionario->preguntas->map(function ($pregunta): array {
144                     return [
145                         'PreguntaID' => $pregunta->PreguntaID,
146                         'Texto' => $pregunta->Texto,
147                         'Tipo' => $pregunta->Tipo,
148                         'respuestas' => $pregunta->respuestas->map(function ($respuesta): array {
149                             return [
150                                 'RespuestaID' => $respuesta->RespuestaID,
151                                 'Texto' => $respuesta->Texto,
152                                 'Valor' => $respuesta->Valor,
153                             ];
154                         })),
155                     ];
156                 })),
157             ];
158         })),
159     ]);
```

Anexo 22 Método para encontrar versiones de cuestionarios

Este método nos permitió publicar un cuestionario asignándolo a un grupo específico, estableciendo fechas de publicación y cierre. Primero valida los datos de entrada para asegurarse de que el grupo y las fechas sean correctos. Luego, busca el cuestionario y el grupo en la base de datos, y si ambos existen, se crea una relación entre ellos con las fechas proporcionadas. Finalmente, responde con un mensaje de éxito y la información del cuestionario y grupo.

```

BackendCEP > app > Http > Controllers > CuestionarioController.php > ...
13 class CuestionarioController extends Controller
162
163
164 //Metodo para publicar un cuestionario
1 reference | 0 overrides
165 public function publicarCuestionario($cuestionarioID, Request $request): JsonResponse|mixed
166 {
167     // Validar entrada
168     $validatedData = $request->validate(rules: [
169         'grupo_id' => 'required|integer|exists:grupos,GrupoID',
170         'fecha_publicacion' => 'required|date',
171         'fecha_cierre' => 'required|date|after_or_equal:fecha_publicacion',
172     ]);
173
174     // Buscar cuestionario y grupo
175     $cuestionario = Cuestionario::find(id: $cuestionarioID);
176     $grupo = Grupos::find(id: $validatedData['grupo_id']);
177
178     if (!$cuestionario || !$grupo) {
179         return response()->json(['message' => 'Cuestionario o grupo no encontrado'], 404);
180     }
181
182     // Asignar el cuestionario al grupo con fechas de publicación y cierre
183     CuestionarioGrupo::create(attributes: [
184         'CuestionarioID' => $cuestionarioID,
185         'GrupoID' => $grupo->GrupoID,
186         'FechaPublicacion' => $validatedData['fecha_publicacion'],
187         'FechaCierre' => $validatedData['fecha_cierre'],
188     ]);
189
190     return response()->json([
191         'message' => 'Cuestionario publicado exitosamente en el grupo especificado',
192         'cuestionario' => $cuestionario,
193         'grupo' => $grupo,
194     ]);
195 }
196

```

Anexo 23 Método para publicar cuestionario

Este método permite al maestro obtener los cuestionarios aplicados a los grupos que ha creado. Primero, recupera los grupos asociados al maestro a través del UsuarioID y luego obtiene los cuestionarios vinculados a esos grupos. Para cada cuestionario, se recopila información relevante como el nombre, la descripción, las fechas de publicación y cierre, además de los datos del grupo y la carrera relacionada. Finalmente, responde con la lista de cuestionarios encontrados o un mensaje de error si ocurre algún problema.

```

BackendCEP > app > Http > Controllers > CuestionarioController.php > ...
13 class CuestionarioController extends Controller
198 // Método para obtener los cuestionarios aplicados para el maestro
199 public function obtenerCuestionariosPorUsuario($usuarioId): JsonResponse|mixed
200 {
201     try {
202         // Obtener los grupos asociados al maestro (grupos creados por el maestro)
203         $gruposMaestro = Grupos::where(column: 'UsuarioID', operator: $usuarioId) // Buscar los grupos creados por el maestro
204         ->pluck(column: 'GrupoID'); // Obtener solo los IDs de los grupos
205
206         // Obtener los cuestionarios asignados a esos grupos junto con sus relaciones
207         $cuestionarios = CuestionarioGrupo::whereIn(column: 'GrupoID', values: $gruposMaestro) // Buscar en la tabla intermedia 'cuestionario_grupo'
208         ->with(relations: ['cuestionario', 'grupo.carrera']) // Cargar las relaciones necesarias
209         ->get();
210
211         // Mapear los resultados de los cuestionarios
212         $resultados = $cuestionarios->map(callback: function (CuestionarioGrupo $cuestionarioGrupo): array {
213             // Formatear la información de cada cuestionario y del grupo al que pertenece
214             return [
215                 'cuestionarioID' => $cuestionarioGrupo->cuestionario->CuestionarioID, // Obtener el ID del cuestionario
216                 'Nombre' => $cuestionarioGrupo->cuestionario->Nombre, // Obtener el nombre del cuestionario
217                 'Descripcion' => $cuestionarioGrupo->cuestionario->Descripcion, // Obtener la descripción del cuestionario
218                 'FechaPublicacion' => $cuestionarioGrupo->FechaPublicacion, // Fecha de publicación desde cuestionario_grupo
219                 'FechaCierre' => $cuestionarioGrupo->FechaCierre, // Fecha de cierre desde cuestionario_grupo
220                 // Datos del grupo
221                 'GrupoID' => $cuestionarioGrupo->grupo->GrupoID, // ID del grupo
222                 'GrupoNombre' => $cuestionarioGrupo->grupo->Nombre, // Nombre del grupo
223                 'GrupoSemestre' => $cuestionarioGrupo->grupo->semestre, // Semestre del grupo
224                 'GrupoCarreraID' => $cuestionarioGrupo->grupo->CarreraID, // Carrera del grupo
225                 // Acceder al nombre de la carrera a través de la relación
226                 'CarreraNombre' => $cuestionarioGrupo->grupo->carrera->Nombre, // Nombre de la carrera asociada al grupo
227             ];
228         });
229

```

Anexo 24 Método para la obtención de cuestionarios por usuario

```

BackendCEP > app > Http > Controllers > CuestionarioController.php > ...
13 class CuestionarioController extends Controller
199 public function obtenerCuestionariosPorUsuario($usuarioId): JsonResponse|mixed
225 // Acceder al nombre de la carrera a través de la relación
226 'CarreraNombre' => $cuestionarioGrupo->grupo->carrera->Nombre, // Nombre de la carrera asociada al grupo
227 ];
228 });
229
230 // Responder con los cuestionarios encontrados
231 return response()->json([
232     'message' => 'Cuestionarios obtenidos para los grupos del usuario',
233     'cuestionarios' => $resultados,
234 ]);
235
236 } catch (\Exception $e) {
237     // Manejo de error en caso de que la consulta falle
238     return response()->json([
239         'message' => 'Error al obtener los cuestionarios',
240         'error' => $e->getMessage(),
241     ], 500);
242 }
243

```

Anexo 25 Respuesta de la obtención del cuestionario

Este método nos permite para obtener la información detallada de un cuestionario específico a partir de su ID. Primero, busca el cuestionario en la base de datos utilizando el identificador proporcionado. Si el cuestionario no existe, devuelve un mensaje de error con el código 404. Si se encuentra, responde con los detalles del cuestionario en formato JSON. En caso de error, se captura la excepción y se devuelve un mensaje de error con el código 500.

```

//Obtener la información de un solo cuestionario
1 reference | 0 overrides
public function getCuestionario($cuestionarioId): JsonResponse|mixed
{
    try {
        $cuestionario = Cuestionario::find(id: $cuestionarioId);

        if (!$cuestionario) {
            return response()->json([
                'success' => false,
                'message' => 'Cuestionario no encontrado.',
            ], 404);
        }

        return response()->json([
            'success' => true,
            'data' => $cuestionario,
        ], 200);
    } catch (\Exception $e) {
        return response()->json([
            'success' => false,
            'message' => 'Error al obtener el cuestionario.',
            'error' => $e->getMessage(),
        ], 500);
    }
}

```

Anexo 26 Método para obtener la información de un solo cuestionario

Este método nos permite duplicar y editar un cuestionario existente, creando una nueva versión con las modificaciones proporcionadas. Primero, valida la entrada del usuario, asegurándose de que los datos del cuestionario y sus preguntas sean correctos. Luego, crea una nueva versión del cuestionario con los datos proporcionados y asocia las preguntas editadas a la nueva versión. Finalmente, responde con un mensaje confirmando la creación de la nueva versión, junto con los detalles del cuestionario original y la nueva versión.

```

BackendCEP > app > Http > Controllers > CuestionarioController.php > ...
13 class CuestionarioController extends Controller
2 references | 0 overrides
273 public function duplicarYeditarCuestionario($cuestionarioID, Request $request): JsonResponse|mixed
274 {
275     // Usar el ID 1 como predeterminado si no se proporciona uno válido
276     if (!$cuestionarioID || $cuestionarioID <= 0) {
277         $cuestionarioID = 1;
278     }
279
280     $cuestionarioOriginal = Cuestionario::with(relations: ['preguntas.respuestas'])->find(id: $cuestionarioID);
281
282     if (!$cuestionarioOriginal) {
283         return response()->json(['message' => 'Cuestionario no encontrado'], 404);
284     }
285
286     // Validar los datos de entrada
287     $validatedData = $request->validate(rules: [
288         'nombre' => 'required|string|max:255',
289         'descripcion' => 'nullable|string',
290         'fecha_publicacion' => 'required|date',
291         'fecha_cierre' => 'required|date|after_or_equal:fecha_publicacion',
292         'preguntas' => 'required|array',
293         'preguntas.*.PreguntaID' => 'exists:preguntas,PreguntaID',
294         'preguntas.*.Texto' => 'required|string',
295         'preguntas.*.Tipo' => 'required|string|in:Positiva,Negativa'
296     ]);
297
298     // Crear la nueva versión del cuestionario
299     $nuevaVersion = Cuestionario::create(attributes: [
300         'GrupoID' => $cuestionarioOriginal->GrupoID,
301         'Nombre' => $validatedData['nombre'],
302         'Descripcion' => $validatedData['descripcion'],
303         'FechaPublicacion' => $validatedData['fecha_publicacion'],
304         'FechaCierre' => $validatedData['fecha_cierre'],
305     ]);

```

Anexo 27 Método para duplicar y editar cuestionarios


```

// Asociar preguntas a la nueva versión
foreach ($validatedData['preguntas'] as $preguntaData) {
    $pregunta = Pregunta::find(id: $preguntaData['PreguntaID']);
    if ($pregunta) {
        // Crear una nueva asociación con la pregunta editada
        CuestionarioPregunta::create(attributes: [
            'CuestionarioID' => $nuevaVersion->CuestionarioID,
            'PreguntaID' => $pregunta->PreguntaID,
        ]);

        // Actualizar el texto y el tipo de la pregunta para la nueva versión
        $pregunta->Texto = $preguntaData['Texto'];
        $pregunta->Tipo = $preguntaData['Tipo'];
        $pregunta->save();
    }
}

return response()->json([
    'message' => 'Nueva versión del cuestionario creada exitosamente',
    'cuestionario_original' => $cuestionarioOriginal,
    'nueva_version' => $nuevaVersion,
]);
}
}

```

Controlador cuestionario resuelto

Este controlador maneja la funcionalidad para marcar un cuestionario como resuelto y obtener los cuestionarios resueltos de un alumno. Al marcar un cuestionario como resuelto, se guarda un registro con los detalles del cuestionario y el alumno. Además, cuando se solicita la lista de cuestionarios resueltos de un alumno y un grupo específicos, se obtienen los cuestionarios resueltos, incluyendo el nombre del cuestionario asociado. Finalmente, la respuesta se estructura de manera que se devuelve el estado y detalles del cuestionario resuelto, asegurando que las relaciones sean válidas

```

BackendCEP > app > Http > Controllers > CuestionarioResueltoController.php > ...
5 use App\Http\Controllers\Controller;
6 use Illuminate\Http\Request;
7 use App\Models\CuestionarioResuelto;
8 use App\Helpers\ResponseHelper;
9 use App\Models\Cuestionario;
10 use App\Models\CuestionarioGrupo;
11 class CuestionarioResueltoController extends Controller
12 {
13     //Método para marcar un cuestionario como resuelto
14     public function marcarComoResuelto(Request $request): JsonResponse|mixed
15     {
16         $resuelto = CuestionarioResuelto::create(attributes: [
17             'UsuarioID' => $request->UsuarioID,
18             'CuestionarioID' => $request->CuestionarioID,
19             'GrupoID' => $request->GrupoID,
20             'FechaResuelto' => now(),
21             'Resuelto' => true,
22         ]);
23         return ResponseHelper::success(data: $resuelto, message: 'Cuestionario marcado como resuelto');
24     }
25
26     // Método para obtener cuestionarios resueltos
27     public function obtenerCuestionariosResueltos($usuarioId, $grupoId): JsonResponse|mixed
28     {
29         // Obtener cuestionarios resueltos con la relación cuestionario
30         $cuestionariosResueltos = CuestionarioResuelto::with(relations: 'cuestionario')
31             ->where(column: 'UsuarioID', operator: $usuarioId)
32             ->where(column: 'GrupoID', operator: $grupoId)
33             ->where(column: 'Resuelto', operator: true)
34             ->get();
35
36         // Mapear los resultados con validación de relaciones
37         $resultados = $cuestionariosResueltos->map(callback: function (CuestionarioResuelto $resuelto): array {

```

Anexo 28 Método para marcar y obtener los cuestionarios resueltos

Este método recupera todos los cuestionarios resueltos por un alumno en distintos grupos, incluyendo detalles del cuestionario y del grupo. Para cada cuestionario resuelto, se extrae información como el nombre, descripción del cuestionario, y fechas de publicación y cierre, si están disponibles. Además, se mapean los resultados para mostrar las relaciones entre los cuestionarios y los grupos, y se aseguran de que, si alguna relación es nula, se devuelve un valor predeterminado. Finalmente, se devuelve una respuesta con los datos mapeados y un mensaje de éxito.

```

BackendCEP > app > Http > Controllers > CuestionarioResueltoController.php > ...
11 class CuestionarioResueltoController extends Controller
12 {
13     public function obtenerCuestionariosResueltos($usuarioId, $grupoId): JsonResponse|mixed
14     {
15         // Mapear los resultados con validación de relaciones
16         $resultados = $cuestionariosResueltos->map(callback: function (CuestionarioResuelto $resuelto): array {
17             $cuestionario = $resuelto->cuestionario; // Relación con cuestionario
18
19             return [
20                 'CuestionarioResueltoID' => $resuelto->CuestionarioResueltoID,
21                 'UsuarioID' => $resuelto->UsuarioID,
22                 'CuestionarioID' => $resuelto->CuestionarioID,
23                 'GrupoID' => $resuelto->GrupoID,
24                 'FechaResuelto' => $resuelto->FechaResuelto,
25                 'Resuelto' => $resuelto->Resuelto,
26                 'Nombre' => $cuestionario?->Nombre ?? 'Cuestionario no disponible', // Validar relación nula
27             ];
28         });
29
30         return ResponseHelper::success(data: $resultados, message: 'Cuestionarios resueltos obtenidos correctamente');
31     }
32
33     // Método para obtener cuestionarios resueltos por alumno
34     public function obtenerCuestionariosResueltosPorAlumno($usuarioId): JsonResponse|mixed
35     {
36         // Obtener todos los cuestionarios resueltos del alumno en diferentes grupos
37         $cuestionariosResueltos = CuestionarioResuelto::with(relations: ['cuestionario.cuestionarioGrupos', 'grupo'])
38             ->where(column: 'UsuarioID', operator: $usuarioId)
39             ->where(column: 'Resuelto', operator: true)
40             ->get();
41
42         // Mapear los datos para incluir la información del cuestionario, grupo y las fechas
43         $resultados = $cuestionariosResueltos->map(callback: function (CuestionarioResuelto $resuelto): array {
44             $cuestionario = $resuelto->cuestionario; // Relación con cuestionario
45             $grupo = $resuelto->grupo; // Relación con grupo

```

Anexo 29 Método para obtener los cuestionarios resueltos por alumno

Este método obtiene los cuestionarios pendientes de un alumno para un grupo específico. Primero, se recopilan los IDs de los cuestionarios que ya han sido resueltos por el alumno en ese grupo. Luego, se obtienen los cuestionarios asignados al grupo, filtrando aquellos que ya han sido resueltos, y se mapean los resultados para incluir información como el nombre, descripción y fechas de los cuestionarios. Finalmente, la función devuelve una respuesta con los cuestionarios pendientes y un mensaje de éxito.

```

BackEndCIP > app > http > Controllers > CuestionarioResueltoController.php > PHP > CuestionarioResueltoController > obtenerCuestionariosResueltosPorAlumno
11 class CuestionarioResueltoController extends Controller
57 public function obtenerCuestionariosResueltosPorAlumno($usuarioId): JsonResponse{mixed
70     // Obtener cuestionario_grupo para las fechas, si existe
71     $cuestionarioGrupo = $cuestionario->cuestionarioGrupos->firstWhere('GrupoID', $resuelto->GrupoID);
72
73     return [
74         'CuestionarioResueltoID' => $resuelto->CuestionarioResueltoID,
75         'CuestionarioID' => $resuelto->CuestionarioID,
76         'GrupoID' => $resuelto->GrupoID,
77         'FechaResuelto' => $resuelto->FechaResuelto,
78         'Resuelto' => $resuelto->Resuelto,
79         'NombreCuestionario' => $cuestionario->Nombre ?? 'Cuestionario no disponible',
80         'DescripcionCuestionario' => $cuestionario->Descripcion ?? 'Sin descripción',
81         'NombreGrupo' => $grupo->Nombre ?? 'Grupo no disponible',
82         'FechaPublicacion' => $cuestionarioGrupo->FechaPublicacion ?? 'Fecha no asignada',
83         'FechaCierre' => $cuestionarioGrupo->FechaCierre ?? 'Fecha no asignada',
84     ];
85 };
86
87 return ResponseHelper::success(data: $resultados, message: 'Cuestionarios resueltos obtenidos correctamente');
88 }
89
90 // Obtener los cuestionarios pendientes del alumno
91 // referencia [0] overrides
92 public function obtenerCuestionariosPendientes($usuarioId, $grupoId): JsonResponse{mixed
93 {
94     // Obtener los ID de los cuestionarios que ya han sido resueltos por el alumno en el grupo
95     $cuestionariosResueltosIds = CuestionarioResuelto::where(column: 'UsuarioID', operator: $usuarioId)
96         ->where(column: 'GrupoID', operator: $grupoId)
97         ->pluck(column: 'CuestionarioID');
98
99     // Obtener los cuestionarios aplicados al grupo desde 'cuestionario_grupo'
100     $cuestionariosPendientes = CuestionarioGrupo::where(column: 'GrupoID', operator: $grupoId)
101         ->whereNotIn(column: 'CuestionarioID', values: $cuestionariosResueltosIds)
102         ->with(relation: 'cuestionario') // Relación con el modelo Cuestionario
103         ->get();

```

Anexo 30 Método para obtener cuestionarios pendientes por alumno

Este método recupera los cuestionarios asignados a un grupo específico, obtenidos a través de la relación en la tabla cuestionario_grupo. Para ello, se consulta por el ID del grupo y se cargan las relaciones con los modelos de cuestionario y grupo. Luego, se mapean los datos para obtener información relevante como el nombre del cuestionario, la descripción, el nombre del grupo y las fechas de publicación y cierre. Finalmente, se devuelve una respuesta con los cuestionarios aplicados en el grupo y un mensaje de éxito.

```

BackEndCIP > app > http > Controllers > CuestionarioResueltoController.php > PHP > CuestionarioResueltoController > obtenerCuestionariosResueltosPorAlumno
11 class CuestionarioResueltoController extends Controller
57 public function obtenerCuestionariosPendientes($usuarioId, $grupoId): JsonResponse{mixed
103     ->get();
104     ->map(callback: function (CuestionarioGrupo $cuestionarioGrupo): array {
105         return [
106             'CuestionarioID' => $cuestionarioGrupo->CuestionarioID,
107             'Nombre' => $cuestionarioGrupo->Cuestionario->Nombre,
108             'Descripcion' => $cuestionarioGrupo->Cuestionario->Descripcion,
109             'FechaPublicacion' => $cuestionarioGrupo->FechaPublicacion,
110             'FechaCierre' => $cuestionarioGrupo->FechaCierre,
111             'GrupoID' => $cuestionarioGrupo->GrupoID,
112         ];
113     });
114
115 return ResponseHelper::success(data: $cuestionariosPendientes, message: 'Cuestionarios pendientes obtenidos correctamente');
116 }
117
118 // referencia [0] overrides
119 public function obtenerCuestionariosAplicadosPorGrupo($grupoId): JsonResponse{mixed
120 {
121     // Obtener los cuestionarios que han sido asignados al grupo a través de la tabla 'cuestionario_grupo'
122     $cuestionarios = CuestionarioGrupo::where(column: 'GrupoID', operator: $grupoId)
123         ->with(relation: ['cuestionario', 'grupo']) // Cargar relaciones con cuestionario y grupo
124         ->get();
125
126     $resultados = $cuestionarios->map(callback: function (CuestionarioGrupo $cuestionarioGrupo): array {
127         return [
128             'CuestionarioID' => $cuestionarioGrupo->Cuestionario->CuestionarioID,
129             'Nombre' => $cuestionarioGrupo->Cuestionario->Nombre,
130             'Descripcion' => $cuestionarioGrupo->Cuestionario->Descripcion,
131             'GrupoNombre' => $cuestionarioGrupo->grupo->Nombre ?? 'Sin nombre',
132             'FechaPublicacion' => $cuestionarioGrupo->FechaPublicacion,
133             'FechaCierre' => $cuestionarioGrupo->FechaCierre,
134         ];
135     });

```

Anexo 31 Método para obtener cuestionarios aplicados por grupo

Controlador generaciones

Este código agregamos y editamos generaciones dentro del sistema. Primero, al agregar una nueva generación, se validan los datos recibidos, asegurándose de que el nombre sea una cadena válida y que el CarreraID exista en la base de datos. Luego, se crea la nueva generación. Para editar, se verifica si la generación existe, y si es así, se validan y actualizan los datos. Finalmente, se devuelve una respuesta de éxito con la información de la generación creada o actualizada.

```
BackendCEP > app > Http > Controllers > GeneracionController.php > ...
5 use App\Http\Controllers\Controller;
6 use Illuminate\Http\Request;
7 use App\Models\Generaciones;
8 use App\Helpers\ResponseHelper;
9 class GeneracionController extends Controller
10 {
11     //Metodo para agregar una nueva generación
12     public function agregarGeneracion(Request $request): JsonResponse|mixed
13     {
14         $validatedData = $request->validate(rules: [
15             'Nombre' => 'required|string|max:255',
16             'CarreraID' => 'required|exists:carreras,CarreraID',
17         ]);
18         $generacion = Generaciones::create(attributes: $validatedData);
19         return ResponseHelper::success(data: $generacion, message: 'Generación creada correctamente', status: 201);
20     }
21
22     // Editar una generación existente
23     public function editarGeneracion(Request $request, $GeneracionID): JsonResponse|mixed
24     {
25         $generacion = Generaciones::find(id: $GeneracionID);
26         if (!$generacion) {
27             return ResponseHelper::error(message: 'Generación no encontrada', status: 404);
28         }
29         $validatedData = $request->validate(rules: [
30             'Nombre' => 'sometimes|required|string|max:255',
31             'CarreraID' => 'sometimes|required|exists:carreras,CarreraID',
32         ]);
33         $generacion->update($validatedData);
34         return ResponseHelper::success(data: $generacion, message: 'Generación actualizada correctamente');
35     }
36 }
```

Anexo 32 Método para agregar y editar una generación

Este código manejamos las operaciones para obtener, eliminar y consultar generaciones. Primero, la función generaciones () obtiene todas las generaciones registradas en la base de datos y las devuelve. Luego, la función generacion(\$GeneracionID) busca una generación específica por su ID, y si no se encuentra, devuelve un error. Finalmente, la función eliminarGeneracion(\$GeneracionID) busca y elimina una generación mediante su ID, y devuelve una respuesta de éxito al completar la operación.

```

BackendCEP > app > Http > Controllers > GeneracionController.php > ...
9 class GeneracionController extends Controller
30
37 // Obtener todas las generaciones
38 1 reference | 0 overrides
38 public function generaciones(): JsonResponse|mixed
39 {
40     $generaciones = Generaciones::all();
41     return ResponseHelper::success(data: $generaciones, message: 'Generaciones obtenidas correctamente');
42 }
43
44 // Obtener una generación específica por ID
45 1 reference | 0 overrides
45 public function generacion($GeneracionID): JsonResponse|mixed
46 {
47     $generacion = Generaciones::find(id: $GeneracionID);
48     if (!$generacion) {
49         return ResponseHelper::error(message: 'Generación no encontrada', status: 404);
50     }
51     return ResponseHelper::success(data: $generacion, message: 'Generación obtenida correctamente');
52 }
53
54 // Eliminar una generación
55 1 reference | 0 overrides
55 public function eliminarGeneracion($GeneracionID): JsonResponse|mixed
56 {
57     $generacion = Generaciones::findOrFail(id: $GeneracionID);
58     $generacion->delete();
59     return ResponseHelper::success(data: null, message: 'Generación eliminada correctamente');
60 }
61 }

```

Anexo 33 Método para obtener y eliminar una generación

Controlador grupo

Con este código proporcionamos las funcionalidades para gestionar los grupos a los que un alumno pertenece. El método `getAlumnoGrupos($id)` obtiene los grupos, la carrera y la institución asociados a un alumno mediante su ID. Si el alumno no tiene grupos asignados o no existe, se devuelve un error. El método `getGrupoByCodigo(Request $request)` busca un grupo por su código de acceso, incluyendo la información de la carrera e institución. Si no se encuentra el grupo, también se devuelve un error.

```

BackendCEP > app > Http > Controllers > GrupoController.php > ...
5 use App\Models\UsuarioGrupo;
6 use App\Models\Grupos;
7 use App\Models\Usuario;
8 use Illuminate\Http\Request;
9 use App\Helpers\ResponseHelper;
10
11 references | 0 implementations
11 class GrupoController extends Controller
12 {
13     1 reference | 0 overrides
13     public function getAlumnoGrupos($id): JsonResponse|mixed
14     {
15         // Se busca al alumno y carga sus grupos, carrera e institución
16         $alumno = Usuario::with(relations: ['grupos.carrera.institucion'])
17             ->find($id);
18
19         if (!$alumno || $alumno->grupos->isEmpty()) {
20             return ResponseHelper::error(message: 'El alumno no tiene grupos asignados o no existe', status: 404);
21         }
22
23         return ResponseHelper::success(data: $alumno->grupos);
24     }
25
26     // Método para obtener un grupo por código de acceso
27     1 reference | 0 overrides
28     public function getGrupoByCodigo(Request $request): JsonResponse|mixed
29     {
30         $codigo = $request->input(key: 'codigo_acceso');
31
32         $grupo = Grupos::where(column: 'codigo_acceso', operator: $codigo)->with(relations: 'carrera.institucion')->first();
33
34         if (!$grupo) {
35             return ResponseHelper::error(message: 'Grupo no encontrado', status: 404);
36         }
37
38         return ResponseHelper::success(data: $grupo, message: 'Grupo encontrado');
39     }
40 }

```

Anexo 34 Método para buscar alumnos y obtener un grupo por el código de acceso

En este código agregamos a un alumno a un grupo utilizando un código de acceso. Primero, valida que el ID del alumno y el código de acceso sean correctos. Luego, verifica si el grupo existe y si el alumno ya pertenece a ese grupo. Si todo es válido, agrega al alumno al grupo creando un nuevo registro en la tabla de relación UsuarioGrupo. Si ocurre un error durante el proceso, se captura y se registra.

```

BackendCEP > app > Http > Controllers > GrupoController.php > ...
11 class GrupoController extends Controller
12 {
13     // Método para agregar un alumno a un grupo
14     1 reference | 0 overrides
15     public function joinGrupo(Request $request): JsonResponse|mixed
16     {
17         $validated = $request->validate(rules: [
18             'UsuarioID' => 'required|exists:usuarios,UsuarioID',
19             'codigo_acceso' => 'required|string|exists:grupos,codigo_acceso'
20         ]);
21
22         $grupo = Grupos::where(column: 'codigo_acceso', operator: $validated['codigo_acceso'])->first();
23
24         if (!$grupo) {
25             return ResponseHelper::error(message: 'Grupo no encontrado', status: 404);
26         }
27
28         // Se verifica si el alumno ya está en el grupo
29         $exists = UsuarioGrupo::where(column: 'UsuarioID', operator: $validated['UsuarioID'])
30             ->where(column: 'GrupoID', operator: $grupo->GrupoID)
31             ->exists();
32
33         if ($exists) {
34             return ResponseHelper::error(message: 'El alumno ya pertenece a este grupo', status: 400);
35         }
36
37         // Añadimos al alumno al grupo
38         try {
39             UsuarioGrupo::create(attributes: [
40                 'UsuarioID' => $validated['UsuarioID'],
41                 'GrupoID' => $grupo->GrupoID,
42                 'CarreraID' => $grupo->carrera->CarreraID,
43                 'InstitucionID' => $grupo->carrera->institucion->InstitucionID
44             ]);
45         } catch (\Exception $e) {
46             \Log::error('Error al agregar alumno al grupo: ' . $e->getMessage());
47             return ResponseHelper::error(message: 'Error al añadir alumno al grupo', status: 500);
48         }
49     }
50 }

```

Anexo 35 Método para agregar un alumno a un grupo

Este código les permite a los maestros crear un nuevo grupo, validando la información proporcionada por el usuario, como el nombre, semestre, generación, carrera y el ID del usuario. Una vez validados los datos, el grupo se crea con esos valores y se genera automáticamente un código de acceso único. Finalmente, el sistema responde con el grupo creado y el código de acceso generado. Este proceso asegura que se cree el grupo con la información correcta y que se asigne un código único para el acceso.

```
BackendCEP > app > Http > Controllers > GrupoController.php > ...
11 class GrupoController extends Controller
42 public function joinGrupo(Request $request): JsonResponse|mixed
75 }
76
77 return ResponseHelper::success(data: 'Alumno añadido al grupo exitosamente');
78 }
79
80 //Metodo de maestros para crear grupos
81 public function crearGrupo(Request $request): JsonResponse|mixed
82 {
83     // Validar los datos recibidos
84     $validated = $request->validate(rules: [
85         'nombre' => 'required|string|max:255',
86         'semestre' => 'required|string|max:10',
87         'generacion_id' => 'required|integer',
88         'carrera_id' => 'required|integer',
89         'usuario_id' => 'required|integer',
90     ]);
91
92     // Crear un nuevo grupo
93     $grupo = Grupos::create(attributes: [
94         'Nombre' => $validated['nombre'],
95         'semestre' => $validated['semestre'],
96         'GeneracionID' => $validated['generacion_id'],
97         'CarreraID' => $validated['carrera_id'],
98         'UsuarioID' => $validated['usuario_id'],
99         // 'codigo_acceso' se genera automáticamente en el modelo
100     ]);
101
102     return ResponseHelper::success(data: $grupo, message: 'Grupo creado exitosamente. Código de acceso: ' . $grupo->codigo_acceso);
103 }
104
```

Anexo 36 Método para crear grupos para el maestro

Aquí verificamos si un usuario existe y si es un maestro antes de proceder. Si el usuario es un maestro, se obtienen los grupos asociados a él, junto con la carrera e institución, y se filtran solo los alumnos dentro de esos grupos. Si el maestro no tiene grupos asignados, se retorna un mensaje indicándolo, y si existe, se devuelven los grupos correspondientes. El código asegura que solo los maestros puedan acceder a sus propios grupos y los detalles relacionados.

```
BackendCEP > app > Http > Controllers > GrupoController.php > ...
11 class GrupoController extends Controller
105 public function obtenerGruposDeMaestro($usuarioId): JsonResponse|mixed
106 {
107     // Verifica si el usuario existe y es un maestro
108     $usuario = Usuario::find(id: $usuarioId);
109
110     if (!$usuario) {
111         return response()->json(['error' => 'Usuario no encontrado'], 404);
112     }
113
114     if (strcasecmp(string1: $usuario->Tipo, string2: 'Maestro') != 0) {
115         return response()->json(['error' => 'El usuario no es un maestro'], 403);
116     }
117
118     // Obtener los grupos donde el usuario es el creador
119     $grupos = Grupos::where(column: 'UsuarioID', operator: $usuarioId) // Verifica que se esté buscando por UsuarioID
120     ->with(relations: ['carrera.institucion', 'usuarios' => function ($query): void {
121         $query->where('Tipo', 'Alumno'); // Filtrar solo a los alumnos
122     }])
123     ->get();
124
125     if ($grupos->isEmpty()) {
126         return response()->json(['message' => 'El maestro no tiene grupos creados.'], 404);
127     }
128
129     return response()->json($grupos);
130 }
```

Anexo 37 Método para obtener grupos del maestro

Este código buscamos a un grupo específico en la base de datos utilizando su ID, cargando además las relaciones con la carrera e institución asociadas. Si el grupo no se encuentra, devuelve un mensaje de error con el código 404. Si el grupo existe, se devuelve la información del grupo en formato JSON. Esta función permite obtener detalles completos de un grupo junto con los datos relacionados.

```
BackendCEP > app > Http > Controllers > GrupoController.php > ...
11 class GrupoController extends Controller
1 reference | 0 overrides
133 public function show($id): JsonResponse|mixed{
134     // Cargar el grupo junto con carrera e institución
135     $grupo = Grupos::with(relations: 'carrera.institucion')->find(id: $id);
136
137     if (!$grupo) {
138         return response()->json(['message' => 'Grupo no encontrado'], 404);
139     }
140
141     return response()->json(['data' => $grupo]);
142
143
144 }
```

Anexo 38 Método para obtener los grupos con su carrera e institución

En este método obtenemos los integrantes de un grupo junto con su información relacionada, como carrera e institución. Primero, busca el grupo por su ID y luego, mapea los usuarios del grupo para incluir detalles completos como nombre, matrícula, semestre, tipo de usuario y su información de carrera e institución. Si el grupo no se encuentra, devuelve un mensaje de error, mientras que, si existe, se devuelve una lista detallada de los integrantes en formato JSON. Este enfoque permite acceder fácilmente a toda la información relevante de cada integrante del grupo.

```
BackendCEP > app > Http > Controllers > GrupoController.php > PHP > GrupoController > getIntegrantes()
11 class GrupoController extends Controller
1 reference | 0 overrides
145 //Metodo para obtener los integrantes del grupo
146 public function getIntegrantes($id): JsonResponse|mixed
147 {
148     // Obtener el grupo junto con los usuarios, carrera e institución
149     $grupo = Grupos::with(relations: ['usuarios.carrera.institucion'])->find(id: $id);
150
151     if (!$grupo) {
152         return response()->json(['message' => 'Grupo no encontrado'], 404);
153     }
154
155     // Mapear los integrantes del grupo para incluir la información de carrera e institución
156     $integrantes = $grupo->usuarios->map(function ($usuario): array {
157         return [
158             'UsuarioID' => $usuario->UsuarioID,
159             'Nombre' => $usuario->Nombre,
160             'Primer_apellido' => $usuario->Primer_apellido,
161             'Segundo_apellido' => $usuario->Segundo_apellido,
162             'Matricula' => $usuario->Matricula,
163             'Semestre' => $usuario->Semestre,
164             'Tipo' => $usuario->Tipo,
165             'Correo' => $usuario->Correo,
166             'Edad' => $usuario->edad,
167             'Genero' => $usuario->genero,
168             'Carrera' => $usuario->carrera ? [
169                 'CarreraID' => $usuario->carrera->CarreraID,
170                 'Nombre' => $usuario->carrera->Nombre,
171                 'Institucion' => $usuario->carrera->institucion ? [
172                     'InstitucionID' => $usuario->carrera->institucion->InstitucionID,
173                     'Nombre' => $usuario->carrera->institucion->Nombre
174                 ] : null,
175             ] : null,
176         ];
177     });
178     return response()->json(['data' => $integrantes]);
179 }
```

Anexo 39 Método para obtener los integrantes del grupo


```

BackendCEP > app > Http > Controllers > GrupoController.php > PHP > GrupoController > getIntegrantes()
11 class GrupoController extends Controller
180
181 // Método para buscar a un alumno por su nombre
1 reference|0 overrides
182 public function buscarAlumnoPorNombre(Request $request): JsonResponse|mixed
183 {
184     $nombre = $request->input(key: 'nombre'); // Nombre, apellidos o nombre completo a buscar
185
186     // Realizar una búsqueda en los campos Nombre, ApellidoPaterno y ApellidoMaterno
187     $alumnos = Usuario::where(column: 'Tipo', operator: 'Alumno')
188         ->where(column: function(Usuario $query) use ($nombre): void {
189             $query->where(column: 'Nombre', operator: 'like', value: '%' . $nombre . '%')
190             ->orWhere(column: 'Primer_apellido', operator: 'like', value: '%' . $nombre . '%')
191             ->orWhere(column: 'Segundo_apellido', operator: 'like', value: '%' . $nombre . '%')
192             ->orWhereRaw(sql: "CONCAT(Nombre, ' ', Primer_apellido, ' ', Segundo_apellido) LIKE ?", bindings: ['%' . $nombre . '%']);
193         })
194         ->with(relations: 'carrera.institucion') // Obtener la información de la carrera e institución
195         ->get();
196
197 if ($alumnos->isEmpty()) {
198     return ResponseHelper::error(message: 'No se encontraron alumnos con ese nombre', status: 404);
199 }
200
201 return ResponseHelper::success(data: $alumnos);
202 }

```

Anexo 40 Método para buscar a los alumnos por su nombre

Este método permite agregar un alumno a un grupo especificado por su ID. Primero, valida que el UsuarioID exista en la base de datos. Luego, busca el grupo al que se desea añadir el alumno; si no se encuentra, retorna un error. Verifica si el alumno ya pertenece al grupo y, si no es así, lo registra en la tabla de relación con detalles como la carrera e institución asociadas al grupo. Si ocurre un error al guardar, se registra en los logs y se informa al cliente con un mensaje de fallo.

```

BackendCEP > app > Http > Controllers > GrupoController.php > PHP > GrupoController > agregarAlumnoAGrupo()
11 class GrupoController extends Controller
204 // Método para agregar un alumno al grupo
1 reference|0 overrides
205 public function agregarAlumnoAGrupo(Request $request, $grupoId): JsonResponse|mixed
206 {
207     $validated = $request->validate(rules: [
208         'UsuarioID' => 'required|exists:usuarios,UsuarioID',
209     ]);
210
211     $grupo = Grupos::find(id: $grupoId);
212
213     if (!$grupo) {
214         return ResponseHelper::error(message: 'Grupo no encontrado', status: 404);
215     }
216
217     // Verificar si el alumno ya está en el grupo
218     $exists = UsuarioGrupo::where(column: 'UsuarioID', operator: $validated['UsuarioID'])
219         ->where(column: 'GrupoID', operator: $grupo->GrupoID)
220         ->exists();
221
222     if ($exists) {
223         return ResponseHelper::error(message: 'El alumno ya pertenece a este grupo', status: 400);
224     }
225
226     // Añadir al alumno al grupo
227     try {
228         UsuarioGrupo::create(attributes: [
229             'UsuarioID' => $validated['UsuarioID'],
230             'GrupoID' => $grupo->GrupoID,
231             'CarreraID' => $grupo->CarreraID,
232             'InstitucionID' => $grupo->carrera->InstitucionID
233         ]);
234     } catch (\Exception $e) {
235         \Log::error('Error al agregar alumno al grupo: ' . $e->getMessage());
236         return ResponseHelper::error(message: 'Error al añadir alumno al grupo', status: 500);
237     }
238     return ResponseHelper::success(data: 'Alumno añadido al grupo exitosamente');

```

Anexo 41 Método para agregar un alumno a un grupo

En este método nos devuelve una lista de todos los grupos registrados en el sistema. Primero, utilizamos el modelo Grupos para obtener todos los registros, incluyendo las relaciones con carrera e institución, si son necesarias. Si no existen grupos, se envía una respuesta indicando que no hay datos disponibles. En caso contrario, nos retorna la información completa de los grupos en formato JSON, lista para ser utilizada por la interfaz u otros sistemas conectados.

```
BackendCEP > app > Http > Controllers > GrupoController.php > PHP > GrupoController > agregarAlumnoAGrupo()
11 class GrupoController extends Controller
239
240     1 reference | 0 overrides
    public function getAllGrupos(): JsonResponse|mixed
241     {
242         // Obtener todos los grupos con relaciones adicionales si las necesitas
243         $grupos = Grupos::with(relations: 'carrera.institucion')->get();
244
245         if ($grupos->isEmpty()) {
246             return response()->json(['message' => 'No hay grupos disponibles.'], 404);
247         }
248
249         return response()->json(['data' => $grupos]);
250     }
251 }
```

Anexo 42 Método para obtener todos los grupos

Controlador pregunta

Este método nos devuelve todas las preguntas que estén en la base de datos junto con sus relaciones asociadas a las variables. Utilizamos el modelo Pregunta con la función with para cargar las relaciones de manera eficiente y luego devuelve los resultados en formato JSON. Esta estructura nos permitió que la información esté lista para ser procesada o utilizada en la interfaz del sistema. Si no hay preguntas, simplemente nos devolverá un array vacío.

```
BackendCEP > app > Http > Controllers > PreguntaController.php > ...
1  <?php
2
3  namespace App\Http\Controllers;
4
5  use App\Http\Controllers\Controller;
6  use Illuminate\Http\Request;
7  use App\Models\Pregunta;
8
9
10     2 references | 0 implementations
    class PreguntaController extends Controller
11     {
12         //Se obtienen las pregunta con su relacion por variables
13         1 reference | 0 overrides
        public function obtenerPreguntas(): JsonResponse|mixed
14         {
15             $preguntas = Pregunta::with(relations: 'variables')->get();
16             return response()->json($preguntas);
17         }
18     }
19 }
```

Anexo 43 Método para obtener las preguntas con sus variables

Controlador puntaje variable

Con este método calculamos los puntajes de diferentes variables basadas en un cuestionario y un usuario específicos. Primero, definimos las preguntas asociadas a cada variable utilizando un array donde los IDs de las preguntas están organizados por variable. Luego, utilizamos un ciclo foreach para recorrer las variables y llama al método `calcularPuntajePorVariable` para obtener el puntaje correspondiente para cada conjunto de preguntas. Finalmente, almacena los resultados en un array `$puntajes`.

```
BackendCEP > app > Http > Controllers > PuntajeVariableController.php > PHP IntelliSense > PuntajeVariableController
2
3 namespace App\Http\Controllers;
4
5 use App\Http\Controllers\Controller;
6 use Illuminate\Http\Request;
7 use Illuminate\Support\Facades\DB;
8
9 class PuntajeVariableController extends Controller
10 {
11     public function obtenerPuntajesPorCuestionario($UsuarioID, $CuestionarioID): JsonResponse|mixed
12     {
13         // IDs de las preguntas por variable
14         $preguntaVariables = [
15             1 => [1, 12, 23, 45, 34],
16             2 => [2, 13, 24, 46, 35],
17             3 => [3, 14, 25, 47, 36],
18             4 => [4, 15, 26, 48, 37],
19             5 => [5, 16, 27, 49, 38],
20             6 => [6, 17, 28, 50, 39],
21             7 => [7, 18, 29, 51, 40],
22             8 => [8, 19, 30, 52, 41],
23             9 => [9, 20, 31, 53, 42],
24             10 => [10, 21, 32, 54, 43],
25             11 => [11, 22, 33, 55, 44]
26         ];
27
28         // Calcular puntajes de las variables
29         $puntajes = [];
30         foreach ($preguntaVariables as $variable => $preguntaIDs) {
31             $puntajes["puntaje_variable_{$variable}"] = $this->calcularPuntajePorVariable(
32                 UsuarioID: $UsuarioID,
33                 CuestionarioID: $CuestionarioID,
34                 preguntaIDs: $preguntaIDs
35             );
36         }
37     }
38 }
```

Anexo 44 Método para calcular los puntajes por variables

Este fragmento de código ajustamos los puntajes del cuestionario aplicando condiciones a la variable 11 (el factor de corrección) y afectando las demás variables (1 a 10). El método `calcularPuntajePorVariable` obtiene los tipos de preguntas y las respuestas enviadas, ajustando los valores según sean positivas o negativas, y suma un valor fijo de 6 al puntaje final. Finalmente, todos los puntajes se guardan en la base de datos.

```

BackendCEP > app > Http > Controllers > PuntajeVariableController.php > PHP IntelliSense > PuntajeVariableController
9  class PuntajeVariableController extends Controller
11 public function obtenerPuntajesPorCuestionario($UsuarioID, $CuestionarioID): JsonResponse|mixed

38 // Calcular puntaje condicional para la variable 11
39 $puntaje_condicional = $this->aplicarCondicionPuntaje(puntaje: $puntajes['puntaje_variable_11']);
40
41 // Ajustar puntajes de las variables 1 a 10 aplicando la condición
42 foreach (range(start: 1, end: 10) as $variable) {
43     $puntajes["puntaje_variable_{$variable}"] -= $puntaje_condicional;
44 }
45
46 // Guardar los puntajes en la base de datos
47 $this->guardarPuntajesEnBD(UsuarioID: $UsuarioID, CuestionarioID: $CuestionarioID, puntajeVariable0: $puntaje_condicional, $puntajes);
48
49 // Retornar solo los puntajes ajustados
50 return response()->json(['puntajes' => $puntajes]);
51 }
52
1 reference
53 private function calcularPuntajePorVariable($UsuarioID, $CuestionarioID, $preguntaIDs): float|int
54 {
55     // Obtener las preguntas con sus tipos (Positiva o Negativa)
56     $preguntas = DB::table('preguntas')
57         ->whereIn(column: 'PreguntaID', values: $preguntaIDs)
58         ->pluck(column: 'Tipo', key: 'PreguntaID');
59
60     $puntaje = 0;
61
62     // calcular el puntaje según el tipo de cada pregunta
63     foreach ($preguntas as $PreguntaID => $Tipo) {
64         $valor = DB::table('respuestas_enviadas')
65             ->where(column: 'UsuarioID', operator: $UsuarioID)
66             ->where(column: 'CuestionarioID', operator: $CuestionarioID)
67             ->where(column: 'PreguntaID', operator: $PreguntaID)
68             ->value(column: 'respuesta_valor');

```

Anexo 45 Método para calcular el puntaje condicional

```

BackendCEP > app > Http > Controllers > PuntajeVariableController.php > PHP IntelliSense > PuntajeVariableController
9  class PuntajeVariableController extends Controller
53 private function calcularPuntajePorVariable($UsuarioID, $CuestionarioID, $preguntaIDs): float|int
54 {
70     if ($valor !== null) {
71         $puntaje += ($Tipo === 'Positiva' ? $valor : -$valor);
72     }
73 }
74
75 // Sumar 6 al puntaje de la variable
76 $puntaje += 6;
77
78 return $puntaje;
79 }
80
1 reference
81 private function aplicarCondicionPuntaje($puntaje): int
82 {
83     if ($puntaje <= 20) {
84         return 0;
85     } elseif ($puntaje <= 22) {
86         return 3;
87     } elseif ($puntaje <= 24) {
88         return 5;
89     } else {
90         return 7;
91     }
92 }
93
94 // Los otros métodos no requieren modificaciones adicionales.
95
1 reference
96 private function guardarPuntajesEnBD($UsuarioID, $CuestionarioID, $puntajeVariable0, $puntajes): void
97 {
98     $grupoID = $this->obtenerGrupoID(UsuarioID: $UsuarioID, CuestionarioID: $CuestionarioID);
99     if (!$grupoID) {
100         throw new \Exception(message: 'No se pudo encontrar el grupo correspondiente.');
```

Anexo 46 Método para aplicar el puntaje condicional

En el método `guardarPuntajesEnBD` almacenamos los puntajes por variable en la base de datos. Primero, verificamos el grupo relacionado con el usuario y cuestionario, y validamos si ya existen registros previos para evitar duplicados. Luego, recorremos los puntajes calculados, asignando cada uno a su respectiva variable en la tabla `puntajes_por_variable`. Finalmente, se guarda el puntaje específico de la variable 11 (Factor de corrección), asegurando que todos los datos estén correctamente registrados.

```
BackendCEP > app > Http > Controllers > PuntajeVariableController.php > PHP IntelliSense > PuntajeVariableController
9  class PuntajeVariableController extends Controller
96  private function guardarPuntajesEnBD($UsuarioID, $CuestionarioID, $puntajeVariable0, $puntajes): void
103      // Verificar si los puntajes ya están guardados
104      $puntajesExistentes = DB::table('puntajes_por_variable')
105          ->where(column: 'UsuarioID', operator: $UsuarioID)
106          ->where(column: 'GrupoID', operator: $grupoID)
107          ->where(column: 'CuestionarioID', operator: $CuestionarioID)
108          ->exists();
109      if ($puntajesExistentes) {
110          // Si ya existen, no realizar nuevas inserciones
111          return;
112      }
113
114      foreach ($puntajes as $variable => $puntaje) {
115          $variableID = $this->obtenerVariableID(variable: $variable);
116          DB::table('puntajes_por_variable')->insert(values: [
117              'UsuarioID' => $UsuarioID,
118              'GrupoID' => $grupoID,
119              'CuestionarioID' => $CuestionarioID,
120              'VariableID' => $variableID,
121              'puntaje' => $puntaje,
122              'created_at' => now(),
123              'updated_at' => now(),
124          ]);
125      }
126
127      // Guardar puntaje de la variable 0
128      DB::table('puntajes_por_variable')->insert(values: [
129          'UsuarioID' => $UsuarioID,
130          'GrupoID' => $grupoID,
131          'CuestionarioID' => $CuestionarioID,
132          'VariableID' => 11,
133          'puntaje' => $puntajeVariable0,
134          'created_at' => now(),
135          'updated_at' => now(),
136      ]);
137  }
```

Anexo 47 Método para guardar los puntajes a la BD

En esta función `obtenerGrupoID` buscamos el ID del grupo relacionado con un usuario y un cuestionario. Utilizamos la unión entre las tablas `grupo_usuario` y `cuestionario_grupo` para identificar el grupo en el que ambos coinciden, devolviendo el valor correspondiente de `GrupoID`. Por otro lado, la función `obtenerVariableID` transforma el nombre de la variable en un número entero extrayendo el dígito del texto `puntaje_variable_X`, lo que facilita su uso en consultas o cálculos.

```

BackendCEP > app > Http > Controllers > PuntajeVariableController.php > PHP IntelliSense > PuntajeVariableController
9 class PuntajeVariableController extends Controller
139 1 reference
140 private function obtenerGrupoID($UsuarioID, $CuestionarioID): mixed
141 {
142     return DB::table('grupo_usuario')
143         ->join(table: 'cuestionario_grupo', first: 'grupo_usuario.GrupoID', operator: '=', second: 'cuestionario_grupo.GrupoID')
144         ->where(column: 'grupo_usuario.UsuarioID', operator: $UsuarioID)
145         ->where(column: 'cuestionario_grupo.CuestionarioID', operator: $CuestionarioID)
146         ->value(column: 'grupo_usuario.GrupoID');
147 }
148 1 reference
149 private function obtenerVariableID($variable): int
150 {
151     return (int)str_replace(search: 'puntaje_variable_', replace: '', subject: $variable);
152 }

```

Anexo 48 Método para obtener los puntajes de un grupo

Controlador respuestas enviadas

En esta función se registran las respuestas enviadas de los alumnos, validándolas previamente para asegurarnos que los datos sean correctos y coincidan con las referencias en las tablas cuestionarios y preguntas. Las respuestas son procesadas mediante un bucle y se guardan en la tabla RespuestaEnviada junto con el UsuarioID. Además, utilizamos el registro de logs para monitorear la información recibida y validar el proceso. Al finalizar, nos devuelve un mensaje de éxito cuando se guardan las respuestas correctamente.

```

BackendCEP > app > Http > Controllers > RespuestaEnviadaController.php > ...
3 namespace App\Http\Controllers;
4
5 use Illuminate\Http\Request;
6 use App\Models\RespuestaEnviada;
7 use Illuminate\Support\Facades\Log;
8
9 2 references | 0 implementations
10 class RespuestaEnviadaController extends Controller
11 {
12 1 reference | 0 overrides
13 public function store(Request $request, $UsuarioID): JsonResponse|mixed
14 {
15     Log::info('Solicitud recibida desde Angular', $request->all());
16     // Validar los datos entrantes
17     $request->validate(rules: [
18         'respuestas' => 'required|array',
19         'respuestas.*.CuestionarioID' => 'required|integer|exists:cuestionarios,CuestionarioID',
20         'respuestas.*.PreguntaID' => 'required|integer|exists:preguntas,PreguntaID',
21         'respuestas.*.respuesta_valor' => 'required|integer',
22     ]);
23     // Registrar respuestas recibidas
24     Log::info('Respuestas recibidas:', $request->input(key: 'respuestas'));
25     // Recorrer las respuestas y guardarlas
26     foreach ($request->input(key: 'respuestas') as $respuesta) {
27         RespuestaEnviada::create(attributes: [
28             'CuestionarioID' => $respuesta['CuestionarioID'],
29             'PreguntaID' => $respuesta['PreguntaID'],
30             'respuesta_valor' => $respuesta['respuesta_valor'],
31             'UsuarioID' => $UsuarioID,
32         ]);
33     }
34
35     return response()->json(['message' => 'Respuestas guardadas correctamente'], 200);
36 }
37 }

```

Anexo 49 Método para registrar las respuestas recibidas

Controlador sumatoria puntajes

En esta función procesamos y guardamos los puntajes por categorías y variables relacionadas con un cuestionario. Primero, obtenemos todas las categorías junto con sus variables y expresiones asociadas. Para cada variable, se calcula su puntaje total desde la tabla `puntajes_por_variable`, y se determina el porcentaje basado en su puntaje máximo posible. Luego, estos datos se almacenan o actualizan en la tabla `sumatoria_puntuaciones`, asegurando que incluyan información de usuario, categoría, variable y expresiones. Finalmente, nos devuelve una respuesta indicando que el proceso se realizó con éxito.

```
BackendCEP > app > Http > Controllers > SumatoriaPuntuacionesController.php > PHP > SumatoriaPuntuacionesController

5 use App\Http\Controllers\Controller;
6 use Illuminate\Http\Request;
7 use App\Models\RespuestaEnviada;
8 use Illuminate\Support\Facades\DB;
9 use App\Models\SumatoriaPuntajes;
10 use App\Models\Variable;
11 use App\Models\Categoria;
12 use App\Models\Expresiones;
13
14
15 7 references | 0 implementations
16 class SumatoriaPuntuacionesController extends Controller
17 {
18     1 reference | 0 overrides
19     public function guardarPuntajesPorCategoriaYVariable($usuarioID, $cuestionarioID): JsonResponse|mixed
20     {
21         // Obtener todas las categorías
22         $categorias = Categoria::with(relations: ['variables.expresiones'])->get();
23
24         foreach ($categorias as $categoria) {
25             foreach ($categoria->variables as $variable) {
26                 // Obtener el puntaje de la variable desde la tabla puntajes_por_variable
27                 $puntajeVariable = DB::table('puntajes_por_variable')
28                     ->where(column: 'UsuarioID', operator: $usuarioID)
29                     ->where(column: 'CuestionarioID', operator: $cuestionarioID)
30                     ->where(column: 'VariableID', operator: $variable->VariableID)
31                     ->value(column: 'puntaje'); // Obtener el valor de la columna "puntaje"
32
33                 // Calcular el puntaje máximo
34                 $puntajeMaximo = DB::table('pregunta_variable')
35                     ->where(column: 'VariableID', operator: $variable->VariableID)
36                     ->count() * 5; // Considerando que cada respuesta tiene un máximo de 5 puntos
37
38                 $porcentaje = $puntajeMaximo > 0 ? ($puntajeVariable / $puntajeMaximo) * 100 : 0;
39
40                 foreach ($variable->expresiones as $expresion) {
41                     // Guardar en la tabla sumatoria_puntajes
42                 }
43             }
44         }
45     }
46 }
```

Anexo 50 Método para guardar puntajes por categorías

En este método obtenemos los puntajes de los alumnos en un grupo específico para un cuestionario determinado. Primero, se recuperan los alumnos del grupo con sus datos básicos, y si no hay alumnos, nos devuelve un mensaje de error. Luego, para cada alumno, se obtienen los puntajes de la tabla `sumatoria_puntuaciones`, que incluyen detalles de las categorías, variables y expresiones asociadas. Finalmente, nos retorna los resultados en formato JSON, con los datos de los alumnos y sus puntajes correspondientes.

```

BackendCEP > app > Http > Controllers > SumatoriaPuntuacionesController.php > PHP > SumatoriaPuntuacionesController
15 class SumatoriaPuntuacionesController extends Controller
16 {
17     1 reference | 0 overrides
18
19     public function obtenerPuntajesPorGrupo($GrupoID, $CuestionarioID): JsonResponse|mixed
20     {
21         // Obtener los alumnos del grupo
22         $alumnos = DB::table('grupo_usuario')
23             ->join(table: 'usuarios', first: 'grupo_usuario.UsuarioID', operator: '=', second: 'usuarios.UsuarioID')
24             ->where(column: 'grupo_usuario.GrupoID', operator: $GrupoID)
25             ->select(columns: 'usuarios.UsuarioID', 'usuarios.Nombre', 'usuarios.Primer_apellido', 'usuarios.Segundo_apellido')
26             ->get();
27
28         if ($alumnos->isEmpty()) {
29             return response()->json(['message' => 'No se encontraron alumnos en este grupo.'], 404);
30         }
31
32         // Obtener los puntajes para cada alumno
33         $resultados = [];
34
35         foreach ($alumnos as $alumno) {
36             $puntajes = SumatoriaPuntajes::with(relations: ['categoria', 'variable', 'expresion'])
37                 ->where(column: 'UsuarioID', operator: $alumno->UsuarioID)
38                 ->where(column: 'CuestionarioID', operator: $CuestionarioID)
39                 ->get();
40
41             $resultados[] = [
42                 'alumno' => [
43                     'id' => $alumno->UsuarioID,
44                     'nombre' => $alumno->Nombre,
45                     'primer_apellido' => $alumno->Primer_apellido,
46                     'segundo_apellido' => $alumno->Segundo_apellido,
47                 ],
48                 'puntajes' => $puntajes,
49             ];
50         }
51
52         return response()->json([
53             'message' => 'Puntajes obtenidos correctamente.',
54         ]);
55     }
56 }

```

Anexo 51 Método para obtener la sumatoria de puntajes por grupo

En este método, obtenemos la información del usuario y del cuestionario a partir de sus respectivos IDs. Luego, nos permite recuperar los puntajes agrupados por categoría desde la base de datos, sumando los puntajes totales y máximos de cada categoría. Posteriormente, calculamos el porcentaje de cada categoría en función del puntaje obtenido sobre el máximo posible. Finalmente, nos devuelve un JSON con los detalles del usuario, el cuestionario y los puntajes por categoría, incluyendo los porcentajes calculados.

```

BackendCEP > app > Http > Controllers > SumatoriaPuntuacionesController.php > PHP > SumatoriaPuntuacionesController
15 class SumatoriaPuntuacionesController extends Controller
16 {
17     public function obtenerPuntajesPorCategoria($UsuarioID, $CuestionarioID): JsonResponse|mixed
18     {
19         // Obtener información del usuario
20         $usuario = DB::table('usuarios')
21             ->where(column: 'UsuarioID', operator: $UsuarioID)
22             ->select(columns: 'Nombre', 'Primer_apellido', 'Segundo_apellido')
23             ->first();
24
25         if (!$usuario) {
26             return response()->json(['message' => 'Usuario no encontrado.'], 404);
27         }
28
29         // Obtener información del cuestionario
30         $cuestionario = DB::table('cuestionarios')
31             ->where(column: 'CuestionarioID', operator: $CuestionarioID)
32             ->select(columns: 'Nombre as cuestionario_nombre', 'Descripcion as cuestionario_descripcion')
33             ->first();
34
35         if (!$cuestionario) {
36             return response()->json(['message' => 'Cuestionario no encontrado.'], 404);
37         }
38
39         // Obtener puntajes agrupados por categoría
40         $categorias = SumatoriaPuntajes::with(relations: 'categoria')
41             ->select(columns: 'CategorialD', DB::raw('SUM(total_puntuacion) as total_puntuacion'), DB::raw('SUM(puntaje_maximo) as puntaje_maximo'))
42             ->where(column: 'UsuarioID', operator: $UsuarioID)
43             ->where(column: 'CuestionarioID', operator: $CuestionarioID)
44             ->groupBy(groups: 'CategorialD')
45             ->get();
46
47         ->map(callback: function (SumatoriaPuntajes $item): SumatoriaPuntajes {
48             $item->porcentaje = $item->puntaje_maximo > 0 ? ($item->total_puntuacion / $item->puntaje_maximo) * 100 : 0;
49             return $item;
50         });
51
52         return response()->json([
53             'message' => 'Puntajes por categoria obtenidos correctamente.',
54         ]);
55     }
56 }

```

Anexo 52 Sumatoria de puntajes por categorías

Este código nos permite obtener los puntajes agrupados por expresiones dentro de una categoría específica de un cuestionario. Primero, obtenemos la información del usuario, el cuestionario y la categoría. Luego, recuperamos los puntajes de las expresiones dentro de la categoría seleccionada, sumando los puntajes y calculando los porcentajes correspondientes. Finalmente, nos devuelve una respuesta JSON con los detalles del usuario, el cuestionario, la categoría y los puntajes por expresión calculados.

```
BackendCEP > app > Http > Controllers > SumatoriaPuntuacionesController.php > PHP > SumatoriaPuntuacionesController > obtenerExpresionesPorCategoría()
15 class SumatoriaPuntuacionesController extends Controller
156 public function obtenerExpresionesPorCategoría($usuarioID, $cuestionarioID, $categoriaID): JsonResponse|mixed
174 if (!$cuestionario) {
175     return response()->json(['message' => 'cuestionario no encontrado.'], 404);
176 }
177 // Obtener información de la categoría
178 $categoria = DB::table('categorias')
179     ->where(column: 'categoriaID', operator: $categoriaID)
180     ->select(columns: 'Nombre as categoria_nombre')
181     ->first();
182
183 if (!$categoria) {
184     return response()->json(['message' => 'Categoría no encontrada.'], 404);
185 }
186
187 // Obtener puntajes agrupados por expresiones dentro de la categoría
188 $expresiones = SumatoriaPuntajes::with('relations:expresion')
189     ->select(columns: 'ExpresionesID', DB::raw('SUM(total_puntuacion) as total_puntuacion'),
190     DB::raw('SUM(puntaje_maximo) as puntaje_maximo'))
191     ->where(column: 'UsuarioID', operator: $usuarioID)
192     ->where(column: 'CuestionarioID', operator: $cuestionarioID)
193     ->where(column: 'CategoríaID', operator: $categoriaID)
194     ->groupBy(groups: 'ExpresionesID')
195     ->get();
196 ->map(callback: function (SumatoriaPuntajes $item): SumatoriaPuntajes {
197     $item->porcentaje = $item->puntaje_maximo > 0 ? ($item->total_puntuacion / $item->puntaje_maximo) * 100 : 0;
198     return $item;
199 });
200
201 return response()->json([
202     'message' => 'Puntajes por categoría obtenidos correctamente.',
203     'usuario' => [
204         'nombre' => $usuario->Nombre,
205         'primer_apellido' => $usuario->Primer_apellido,
206         'segundo_apellido' => $usuario->Segundo_apellido,
207     ],
208     'cuestionario' => [
209         'nombre' => $cuestionario->cuestionario_nombre,
210         'descripcion' => $cuestionario->cuestionario_descripcion,
211     ],
212     'categoria' => [
213         'nombre' => $categoria->categoria_nombre,
214     ],
215     'data' => $expresiones,
216 ], 200);
217 }
218
219 // Obtener las preguntas por categoría, expresión
220 reference|O overrides
221 public function obtenerPreguntasPorExpresion($usuarioID, $cuestionarioID, $expresionesID): JsonResponse|mixed
222 {
223     // Obtener información del usuario
224     $usuario = DB::table('usuarios')
225         ->where(column: 'UsuarioID', operator: $usuarioID)
226         ->select(columns: 'Nombre', 'Primer_apellido', 'Segundo_apellido')
227         ->first();
228
229     if (!$usuario) {
230         return response()->json(['message' => 'Usuario no encontrado.'], 404);
231     }
232
233     // Obtener información del cuestionario
234     $cuestionario = DB::table('cuestionarios')
235         ->where(column: 'CuestionarioID', operator: $cuestionarioID)
236         ->select(columns: 'Nombre as cuestionario_nombre', 'Descripcion as cuestionario_descripcion')
237         ->first();
238
239     if (!$cuestionario) {
240         return response()->json(['message' => 'Cuestionario no encontrado.'], 404);
241     }
242 }
```

Anexo 53 Obtención de la sumatoria de puntajes por expresiones

```
BackendCEP > app > Http > Controllers > SumatoriaPuntuacionesController.php > PHP > SumatoriaPuntuacionesController > obtenerExpresionesPorCategoría()
15 class SumatoriaPuntuacionesController extends Controller
156 public function obtenerExpresionesPorCategoría($usuarioID, $cuestionarioID, $categoriaID): JsonResponse|mixed
208     'cuestionario' => [
209         'nombre' => $cuestionario->cuestionario_nombre,
210         'descripcion' => $cuestionario->cuestionario_descripcion,
211     ],
212     'categoria' => [
213         'nombre' => $categoria->categoria_nombre,
214     ],
215     'data' => $expresiones,
216 ], 200);
217 }
218
219 // Obtener las preguntas por categoría, expresión
220 reference|O overrides
221 public function obtenerPreguntasPorExpresion($usuarioID, $cuestionarioID, $expresionesID): JsonResponse|mixed
222 {
223     // Obtener información del usuario
224     $usuario = DB::table('usuarios')
225         ->where(column: 'UsuarioID', operator: $usuarioID)
226         ->select(columns: 'Nombre', 'Primer_apellido', 'Segundo_apellido')
227         ->first();
228
229     if (!$usuario) {
230         return response()->json(['message' => 'Usuario no encontrado.'], 404);
231     }
232
233     // Obtener información del cuestionario
234     $cuestionario = DB::table('cuestionarios')
235         ->where(column: 'CuestionarioID', operator: $cuestionarioID)
236         ->select(columns: 'Nombre as cuestionario_nombre', 'Descripcion as cuestionario_descripcion')
237         ->first();
238
239     if (!$cuestionario) {
240         return response()->json(['message' => 'Cuestionario no encontrado.'], 404);
241     }
242 }
```

Anexo 54 Obtención de puntajes de preguntas por expresión

En esta función obtenemos las preguntas asociadas a una expresión dentro de un cuestionario. Primero, obtenemos la información del usuario, cuestionario, expresión y categoría relacionada. Luego, recuperamos las preguntas que están vinculadas a esa expresión, así como las respuestas enviadas por el usuario. Finalmente, nos devuelve una respuesta JSON con los detalles del usuario, cuestionario, categoría, expresión y las preguntas con su puntaje.

```
BackendCEP > app > Http > Controllers > SumatoriaPuntuacionesController.php > PHP > SumatoriaPuntuacionesController > obtenerPreguntasPorExpresion()
15 class SumatoriaPuntuacionesController extends Controller
221 public function obtenerPreguntasPorExpresion($UsuarioID, $CuestionarioID, $ExpresionesID): JsonResponse|mixed
239     if (!$cuestionario) {
240         return response()->json(['message' => 'Cuestionario no encontrado.'], 404);
241     }
242
243     // Obtener información de la expresión
244     $expresion = DB::table('expresiones')
245         ->where(column: 'ExpresionesID', operator: $ExpresionesID)
246         ->select(columns: 'Nombre as expresion_nombre', 'VariableID')
247         ->first();
248
249     if (!$expresion) {
250         return response()->json(['message' => 'Expresión no encontrada.'], 404);
251     }
252
253     // Obtener información de la categoría asociada
254     $categoria = DB::table('categorias')
255         ->join(table: 'variables', first: 'categorias.CategoriaID', operator: '=', second: 'variables.CategoriaID')
256         ->where(column: 'variables.VariableID', operator: $expresion->VariableID)
257         ->select(columns: 'categorias.Nombre as categoria_nombre')
258         ->first();
259
260     if (!$categoria) {
261         return response()->json(['message' => 'Categoría no encontrada.'], 404);
262     }
263
264     // Obtener preguntas relacionadas a la expresión
265     $preguntas = DB::table('respuestas_enviadas')
266         ->join(table: 'preguntas', first: 'respuestas_enviadas.PreguntaID', operator: '=', second: 'preguntas.PreguntaID')
267         ->where(column: 'respuestas_enviadas.UsuarioID', operator: $UsuarioID)
268         ->where(column: 'respuestas_enviadas.CuestionarioID', operator: $CuestionarioID)
269         ->whereIn(column: 'respuestas_enviadas.PreguntaID', values: function ($query) use ($expresion): void {
270             $query->select('PreguntaID')
271                 ->from('pregunta_variable')
272                 ->where('VariableID', $expresion->VariableID);
273         })
```

Anexo 55 Obtención de información de las preguntas, categorías y expresiones

```
BackendCEP > app > Http > Controllers > SumatoriaPuntuacionesController.php > PHP > SumatoriaPuntuacionesController > obtenerPreguntasPorExpresion()
15 class SumatoriaPuntuacionesController extends Controller
221 public function obtenerPreguntasPorExpresion($UsuarioID, $CuestionarioID, $ExpresionesID): JsonResponse|mixed
274     ->select(columns: 'preguntas.Texto as pregunta', 'respuestas_enviadas.respuesta_valor as puntaje',
275         DB::raw('5 as puntaje_maximo')) // Puntaje máximo fijo
276     ->get();
277
278     return response()->json([
279         'message' => 'Preguntas por expresión obtenidas correctamente.',
280         'usuario' => [
281             'nombre' => $usuario->Nombre,
282             'primer_apellido' => $usuario->Primer_apellido,
283             'segundo_apellido' => $usuario->Segundo_apellido,
284         ],
285         'cuestionario' => [
286             'nombre' => $cuestionario->cuestionario_nombre,
287             'descripcion' => $cuestionario->cuestionario_descripcion,
288         ],
289         'categoria' => [
290             'nombre' => $categoria->categoria_nombre,
291         ],
292         'expresion' => [
293             'nombre' => $expresion->expresion_nombre,
294         ],
295         'data' => $preguntas,
296     ], 200);
297 }
298
```

Anexo 56 Respuesta de la obtención de los puntajes

Esta parte del código nos permite obtener los puntajes de un usuario para un cuestionario específico. Primero, obtenemos los puntajes del usuario mediante la relación con categorías, variables y expresiones asociadas al cuestionario. Si no se encuentran puntajes, nos devuelve un mensaje indicando que no se encontraron resultados. Si los puntajes son encontrados, nos devuelve una respuesta JSON con los datos de los puntajes obtenidos.

```
reference | 0 overrides
public function obtenerPuntajes($UsuarioID, $CuestionarioID): JsonResponse|mixed
{
    // Obtener los puntajes del usuario para el cuestionario específico
    $puntajes = SumatoriaPuntajes::with(relations: ['categoria', 'variable', 'expresion'])
        ->where(column: 'UsuarioID', operator: $UsuarioID)
        ->where(column: 'CuestionarioID', operator: $CuestionarioID)
        ->get();

    // Verificar si se encontraron puntajes
    if ($puntajes->isEmpty()) {
        return response()->json(['message' => 'No se encontraron puntajes para este usuario y cuestionario.'], 404);
    }

    // Devolver los puntajes obtenidos
    return response()->json([
        'message' => 'Puntajes obtenidos correctamente.',
        'data' => $puntajes,
    ], 200);
}
```

Anexo 57 Obtención de puntajes por usuario y cuestionario

Controlador estadísticos grupales

Este código nos permite calcular los estadísticos de las puntuaciones de un grupo en un cuestionario específico. Obtenemos los valores promedio, desviación estándar, mínimo y máximo de las puntuaciones agrupadas por categoría y variable para los usuarios del grupo. Si no se encuentran datos, nos devuelve un mensaje indicando que no se encontraron resultados. Si los cálculos son exitosos, nos devuelve los estadísticos correspondientes en formato JSON.

```

BackendCEP > app > Http > Controllers > EstadisticosGrupalesController.php > PHP IntelliSense > EstadisticosGrupalesController
5 use App\Http\Controllers\Controller;
6 use Illuminate\Http\Request;
7 use Illuminate\Support\Facades\DB;
8
9 3 references | 0 implementations
9 class EstadisticosGrupalesController extends Controller
10 {
11     1 reference | 0 overrides
11     public function calcularEstadisticosPorGrupo($GrupoID, $CuestionarioID): JsonResponse|mixed
12     {
13         $estadisticos = DB::table('sumatoria_puntuaciones')
14             ->join(table: 'grupo_usuario', first: 'sumatoria_puntuaciones.UsuarioID', operator: '=', second: 'grupo_usuario.UsuarioID')
15             ->where(column: 'grupo_usuario.GrupoID', operator: $GrupoID)
16             ->where(column: 'sumatoria_puntuaciones.CuestionarioID', operator: $CuestionarioID)
17             ->select(
18                 columns: 'sumatoria_puntuaciones.CategoriaID',
19                 'sumatoria_puntuaciones.VariableID',
20                 DB::raw('AVG(total_puntuacion) as promedio'),
21                 DB::raw('STD(total_puntuacion) as desviacion_estandar'),
22                 DB::raw('MIN(total_puntuacion) as minimo'),
23                 DB::raw('MAX(total_puntuacion) as maximo')
24             )
25             ->groupBy(groups: 'sumatoria_puntuaciones.CategoriaID', 'sumatoria_puntuaciones.VariableID')
26             ->get();
27
28         if ($estadisticos->isEmpty()) {
29             return response()->json(['message' => 'No se encontraron datos para el grupo y cuestionario especificados.'], 404);
30         }
31
32         return response()->json([
33             'message' => 'Estadisticos calculados correctamente.',
34             'data' => $estadisticos,
35         ], 200);
36     }
37 }

```

Anexo 58 Calculo de estadístico por grupo

Aquí calculamos los cuartiles de las puntuaciones de un grupo en un cuestionario, obteniendo los datos ordenados de puntuación por categoría y variable. Nos devuelve los resultados con los cuartiles, promedio, desviación estándar, mínimo y máximo de las puntuaciones para cada grupo de categoría y variable. Si no se encuentran datos, nos devuelve un mensaje de error. Los cuartiles se calculan utilizando funciones auxiliares para obtener los valores de los cuartiles 1, 2 (mediana) y 3.

```

BackendCEP > app > Http > Controllers > EstadisticosGrupalesController.php > PHP IntelliSense > EstadisticosGrupalesController
9 class EstadisticosGrupalesController extends Controller
10 {
11     public function calcularCuartiles($GrupoID, $CuestionarioID): JsonResponse|mixed
12     {
13         $datos = DB::table('sumatoria_puntuaciones')
14             ->join(table: 'grupo_usuario', first: 'sumatoria_puntuaciones.UsuarioID', operator: '=', second: 'grupo_usuario.UsuarioID')
15             ->where(column: 'grupo_usuario.GrupoID', operator: '$GrupoID')
16             ->where(column: 'sumatoria_puntuaciones.CuestionarioID', operator: '$CuestionarioID')
17             ->select(columns: 'categoriaID', 'VariableID', 'total_puntuacion')
18             ->orderBy(column: 'total_puntuacion')
19             ->get();
20
21         if ($datos->isEmpty()) {
22             return response()->json(['message' => 'No se encontraron datos para calcular los cuartiles.'], 404);
23         }
24
25         $resultados = [];
26
27         foreach ($datos->groupBy(['CategoriaID', 'VariableID']) as $categoriaVariable => $valores) {
28             $puntuaciones = $valores->pluck('total_puntuacion')->sort()->values();
29             $total = $puntuaciones->count();
30
31             if ($total > 1) {
32                 $resultados[$categoriaVariable] = [
33                     'promedio' => $puntuaciones->avg(),
34                     'desviacion_estandar' => $this->calcularDesviacionEstandar(valores: $puntuaciones),
35                     'minimo' => $puntuaciones->first(),
36                     'maximo' => $puntuaciones->last(),
37                     'cuartil_1' => $this->calcularCuartil(valores: $puntuaciones, percentil: 0.25),
38                     'mediana' => $this->calcularCuartil(valores: $puntuaciones, percentil: 0.50),
39                     'cuartil_3' => $this->calcularCuartil(valores: $puntuaciones, percentil: 0.75),
40                 ];
41             } else {
42                 $resultados[$categoriaVariable] = [
43                     'promedio' => $total > 0 ? $puntuaciones->avg() : null,
44                     'desviacion_estandar' => null,
45                     'minimo' => $puntuaciones->first(),
46                 ];
47             }
48         }
49     }
50 }

```

Anexo 59 Calculo de cuartiles por grupo

Esta función del código nos permite calcular la desviación estándar de un conjunto de valores. Primero, obtenemos la cantidad de valores, y si es mayor a uno, calculamos el promedio de esos valores. Luego, sumamos las diferencias al cuadrado entre cada valor y el promedio, y finalmente, nos devuelve la raíz cuadrada de la media de esas diferencias divididas entre el número de valores menos uno. Si hay menos de dos valores, nos devuelve null, ya que no es posible calcular la desviación estándar.

```

83
84 1 reference
85 private function calcularDesviacionEstandar($valores): float|null
86 {
87     $n = $valores->count();
88     if ($n <= 1) {
89         return null; // No se puede calcular la desviación estándar con menos de 2 valores
90     }
91
92     $promedio = $valores->avg();
93     $suma = $valores->reduce(function ($carry, $item) use ($promedio): float|int {
94         return $carry + pow(num: $item - $promedio, exponent: 2);
95     });
96
97     return sqrt(num: $suma / ($n - 1));
98 }

```

Anexo 60 Calculo de la desviación estándar

Este código nos permite calcular el cuartil de un conjunto de valores, basado en un percentil específico. Primero, obtenemos el tamaño del conjunto de datos y calculamos el índice correspondiente al percentil. Luego, si el índice tiene valores enteros iguales, nos devuelve directamente el valor correspondiente. Si no, interpolamos entre los dos valores cercanos, calculando el valor exacto en el cuartil solicitado y devolviéndolo. Si el conjunto de datos está vacío, nos devuelve null.

```
BackendCEP > app > Http > Controllers > EstadisticosGrupalesController.php > PHP IntelliSense > EstadisticosGrupalesController
9 class EstadisticosGrupalesController extends Controller
100
101 3 references
102 private function calcularCuartil($valores, $percentil): mixed
103 {
104     $n = $valores->count();
105     if ($n == 0) {
106         return null; // No se puede calcular cuartiles con un conjunto vacío
107     }
108     $index = $percentil * ($n - 1);
109     $lower = (int) floor(num: $index);
110     $upper = (int) ceil(num: $index);
111     if ($lower == $upper) {
112         return $valores[$lower];
113     }
114     $lowerValue = $valores[$lower];
115     $upperValue = $valores[$upper];
116     return $lowerValue + ($upperValue - $lowerValue) * ($index - $lower);
117 }
118
119 }
```

Anexo 61 Calculo de cuartil

Routes

Los routers en Laravel son responsables de definir las rutas y los métodos que el sistema utilizará para procesar las solicitudes HTTP. Cada ruta está vinculada a un controlador que ejecuta la lógica específica. Por ejemplo, una ruta como `/cuestionarios/resueltos/{usuarioId}` es gestionada por un controlador que devuelve los cuestionarios resueltos por un alumno. Los routers permiten una comunicación clara y estructurada entre las solicitudes del usuario y la base de datos, facilitando la obtención y manipulación de datos de manera eficiente.

```
BackendCEP > routes > web.php
1  <?php
2
3  use App\Http\Controllers\PuntajePorCuestionarioController;
4  use Illuminate\Support\Facades\Route;
5  use App\Http\Controllers\CuestionarioController;
6  use App\Http\Controllers\InstitucionesController;
7  use App\Http\Controllers\PaísesController;
8  use App\Http\Controllers\RespuestaEnviadaController;
9  use App\Http\Controllers\PuntajeController;
10 use App\Http\Controllers\UsuarioController;
11 use App\Http\Controllers\CarrerasController;
12 use App\Http\Controllers\GrupoController;
13 use App\Http\Controllers\CuestionarioResueltoController;
14 use App\Http\Controllers\PreguntaController;
15 use App\Http\Controllers\GeneracionController;
16 use App\Http\Controllers\SumatoriaPuntuacionesController;
17 use App\Http\Controllers\PuntajeVariableController;
18 use App\Http\Controllers\EstadisticosGrupalesController;
19
20 // Cuestionarios
21 Route::get('/cuestionarios/{cuestionarioID}/preguntas', [CuestionarioController::class, 'getPreguntasRespuestas']);
22 Route::get('/preguntas', [PreguntaController::class, 'obtenerPreguntas']);
23 Route::post('/respuestas/{UsuarioID}/guardarRespuestas', [RespuestaEnviadaController::class, 'store']);
24
25
26 //borrar esto (es del primer cuestionario del home)
27 Route::get('/puntajes/{UsuarioID}', [PuntajeController::class, 'obtenerPuntajesVariables']);
28
29
30 //Usuarios
31 Route::post('/usuarios/register', [UsuarioController::class, 'register']);
32 Route::put('/usuarios/actualizar/{id}', [UsuarioController::class, 'update']);
33 Route::delete('/usuarios/eliminar/{id}', [UsuarioController::class, 'eliminar']);
34 Route::get('/usuarios/alumnos', [UsuarioController::class, 'obtenerAlumnos']);
35 Route::get('/usuarios/maestros', [UsuarioController::class, 'obtenerMaestros']);
36 Route::get('/usuarios/administradores', [UsuarioController::class, 'obtenerAdministradores']);
37 Route::get('/usuarios/all', [UsuarioController::class, 'usuarios']);
```

Anexo 62 Routers

```
BackendCEP > routes > web.php
1  <?php
2
38 Route::get('/usuarios/alumnos/{id}', [UsuarioController::class, 'show']);
39 Route::get('/usuarios/administrador/{id}', [UsuarioController::class, 'show']);
40 Route::get('/usuarios/maestro/{id}', [UsuarioController::class, 'obtenerMaestro']);
41 Route::get('/usuarios/maestros/institucion/{institucionID}', [UsuarioController::class, 'buscarMaestroPorInstitucion']);
42 Route::get('/usuarios/alumnos/institucion/{institucionID}', [UsuarioController::class, 'buscarAlumnoPorInstitucion']);
43 Route::post('/login', [UsuarioController::class, 'login']);
44
45
46 //Instituciones
47 Route::post('catalogs/añadirInstitucion', [InstitucionesController::class, 'crearInstitucion']);
48 Route::get('catalogs/ainstituciones', [InstitucionesController::class, 'instituciones']);
49 Route::get('catalogs/idinstituciones/{id}', [InstitucionesController::class, 'institucion']);
50 Route::get('catalogs/instituciones', [InstitucionesController::class, 'institucionesPorPais']);
51 Route::delete('catalogs/deleteInstitucion/{id}', [InstitucionesController::class, 'destroy']);
52 Route::put('catalogs/instituciones/{id}/actualizar', [InstitucionesController::class, 'actualizarInstitucion']);
53
54 //Países
55 Route::post('catalogs/agregarpais', [PaísesController::class, 'agregarPais']);
56 Route::put('catalogs/paises/actualizar/{id}', [PaísesController::class, 'actualizarPais']);
57 Route::delete('catalogs/paises/eliminar/{id}', [PaísesController::class, 'eliminarPais']);
58 Route::get('catalogs/paises', [PaísesController::class, 'paises']);
59 Route::get('catalogs/paises/{id}', [PaísesController::class, 'pais']);
60 Route::get('catalogs/paises/buscar', [PaísesController::class, 'buscarPaíses']);
61
62
63 //Carreras
64 Route::post('catalogs/carrera/agregar', [CarrerasController::class, 'agregarCarrera']);
65 Route::put('catalogs/carrera/editar/{id}', [CarrerasController::class, 'editarCarrera']);
66 Route::get('catalogs/acarreras', [CarrerasController::class, 'carreras']);
67 Route::get('catalogs/carreras/{id}', [CarrerasController::class, 'carrera']);
68 Route::get('catalogs/carreras', [CarrerasController::class, 'carrerasPorInstitucion']);
69 Route::delete('catalogs/carreras/eliminar/{id}', [CarrerasController::class, 'eliminar']);
70
71 //Generacion
72 Route::post('catalogs/generaciones/agregar', [GeneracionController::class, 'agregarGeneracion']);
73 Route::put('catalogs/generaciones/{id}', [GeneracionController::class, 'editarGeneracion']);
```

Anexo 63 Routers

BackendCEP > routes > web.php

```
1 <?php
71 //Generacion
72 Route::post('catalogs/generaciones/agregar', [GeneracionController::class, 'agregarGeneracion']);
73 Route::put('catalogs/generaciones/{id}', [GeneracionController::class, 'editarGeneracion']);
74 Route::get('catalogs/generaciones', [GeneracionController::class, 'generaciones']);
75 Route::get('catalogs/generaciones/{id}', [GeneracionController::class, 'generacion']);
76 Route::delete('catalogs/generaciones/eliminar/{id}', [GeneracionController::class, 'eliminarGeneracion']);
77
78 //Grupos
79 Route::get('/alumno/{id}/grupos', [GrupoController::class, 'getAlumnoGrupos']);
80 Route::get('/grupos', [GrupoController::class, 'getAllGrupos']);
81
82
83 // Rutas para gestionar el acceso a grupos
84 Route::post('/grupos/buscar-por-codigo', [GrupoController::class, 'getGrupoByCodigo']);
85 Route::post('/grupos/unirse', [GrupoController::class, 'joinGrupo']);
86
87 //Cuestionarios Resueltos
88 Route::post('/cuestionarios/marcar-resuelto', [CuestionarioResueltoController::class, 'marcarComoResuelto']);
89 Route::get('/cuestionarios/resueltos/{usuarioId}/{grupoId}', [CuestionarioResueltoController::class, 'obtenerCuestionariosResueltos']);
90 Route::get('/cuestionarios-resueltos/alumno/{usuarioId}', [CuestionarioResueltoController::class, 'obtenerCuestionariosResueltosPorAlumno']);
91 Route::get('/cuestionarios/{cuestionarioId}', [CuestionarioController::class, 'getCuestionario']);
92
93 //cuestionarios pendientes
94 Route::get('/cuestionarios/pendientes/{usuarioId}/{grupoId}', [CuestionarioResueltoController::class, 'obtenerCuestionariosPendientes']);
95
96 //Maestro
97 //Crear grupo
98 Route::post('/crear-grupos', [GrupoController::class, 'crearGrupo']);
99 Route::get('/maestro/{usuarioId}/grupos', [GrupoController::class, 'obtenerGruposDeMaestro']);
100 Route::get('/grupos/{id}', [GrupoController::class, 'show']);
101 Route::get('/grupos/{id}/integrantes', [GrupoController::class, 'getIntegrantes']);
102 //agregar alumnos
103 Route::get('/alumno/buscar', [GrupoController::class, 'buscarAlumnoPorNombre']);
104 Route::post('/grupos/{grupoId}/alumnos/agregar', [GrupoController::class, 'agregarAlumnoAGrupo']);
```

Anexo 64 Routers

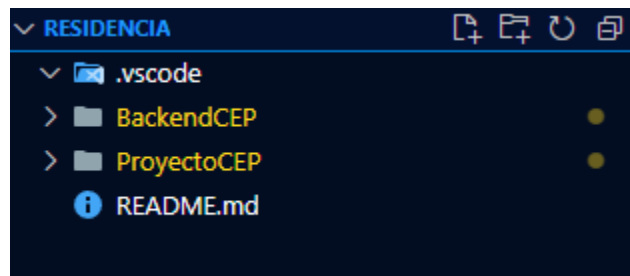
BackendCEP > routes > web.php

```
1 <?php
106 //cuestionarios
107 Route::get('/cuestionarios-aplicados/grupo/{grupoId}', [CuestionarioResueltoController::class, 'obtenerCuestionariosAplicadosPorGrupo']);
108 Route::get('/cuestionarios/usuario/{usuarioId}', [CuestionarioController::class, 'obtenerCuestionariosPorUsuario']);
109
110 Route::post('/cuestionarios/{cuestionarioId}/duplicar-editar', [CuestionarioController::class, 'duplicarYEditarCuestionario']);
111
112
113 //Admin
114 //Crear nuevo cuestionario
115 Route::post('/cuestionarios', [CuestionarioController::class, 'crearCuestionario']);
116 Route::get('/cuestionarios', [CuestionarioController::class, 'getQuizVersions']);
117 Route::post('/cuestionarios/{cuestionarioId}/publicar', [CuestionarioController::class, 'publicarCuestionario']);
118 Route::post('/cuestionarios/{cuestionarioId}/duplicar', [CuestionarioController::class, 'duplicarYEditarCuestionario']);
119 Route::get('/cuestionarios/versions', [CuestionarioController::class, 'getQuizVersions']);
120
121 //Puntuaciones por variable
122 Route::post('/puntajes/sumar-puntuaciones/{UsuarioID}/{CuestionarioID}', [SumatoriaPuntuacionesController::class,
123 'guardarPuntajesPorCategoriaYVariable']);
124 Route::get('/puntajes/sumar-puntuaciones/{UsuarioID}/{CuestionarioID}', [SumatoriaPuntuacionesController::class, 'obtenerPuntajes']);
125 //Para el maestro(sumatoria por grupo)
126 Route::get('/puntajes/consultar/{GrupoID}/{CuestionarioID}', [SumatoriaPuntuacionesController::class, 'obtenerPuntajesPorGrupo']);
127 Route::get('/puntajes/categorias/{UsuarioID}/{CuestionarioID}', [SumatoriaPuntuacionesController::class, 'obtenerPuntajesPorCategoria']);
128 Route::get('/puntajes/expresiones/{UsuarioID}/{CuestionarioID}/{CategoriaID}', [SumatoriaPuntuacionesController::class,
129 'obtenerExpresionesPorCategoria']);
130 Route::get('/puntajes/preguntas/{UsuarioID}/{CuestionarioID}/{ExpresionesID}', [SumatoriaPuntuacionesController::class,
131 'obtenerPreguntasPorExpresion']);
132
133 //Puntuaciones generales guardadas
134 Route::get('/puntajes/cuestionario/{UsuarioID}/{cuestionarioID}', [PuntajeVariableController::class, 'obtenerPuntajesPorCuestionario']);
135
136 //Estadisticos grupales
137 Route::get('/grupos/{GrupoID}/estadisticos/{CuestionarioID}', [EstadisticosGrupalesController::class, 'calcularEstadisticosPorGrupo']);
138 Route::get('/grupos/{GrupoID}/cuartiles/{CuestionarioID}', [EstadisticosGrupalesController::class, 'calcularCuartiles']);
```

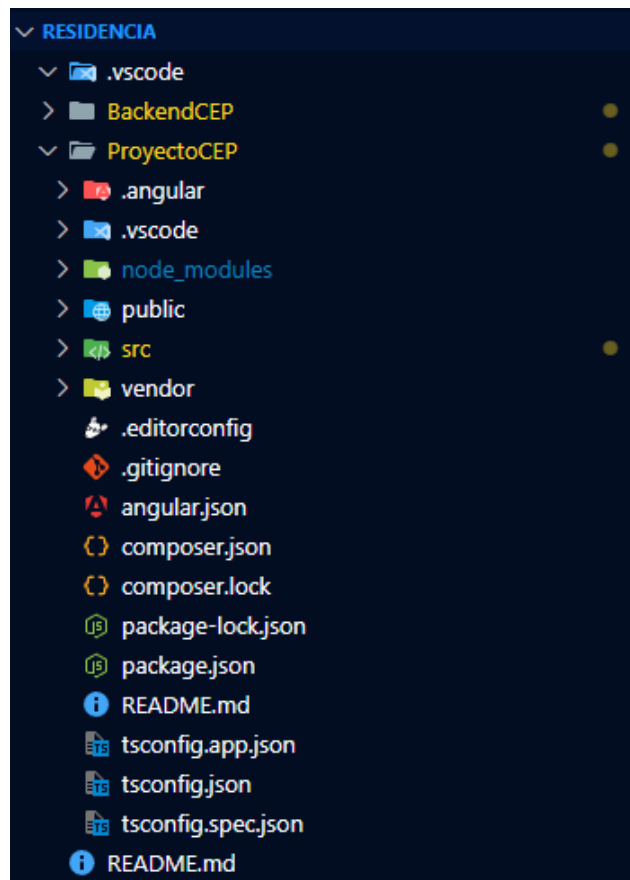
Anexo 65 Routers

Frontend

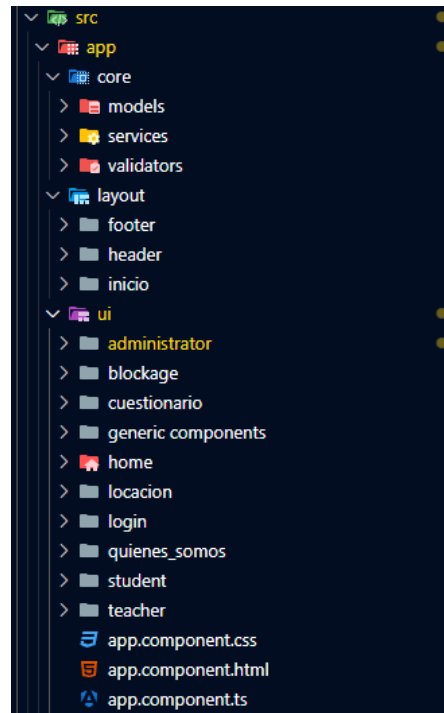
El frontend de este sistema está compuesto por diversas partes que trabajan juntas para proporcionar una interfaz de usuario dinámica y funcional. Utilizando Angular, se crearon componentes que gestionan distintas secciones del sistema, como los cuestionarios, grupos, resultados y el perfil del alumno, administrador y maestro. Estos componentes interactúan con los servicios, los cuales se encargan de hacer solicitudes HTTP al backend para obtener o enviar datos, como los resultados de los cuestionarios o la información de los grupos a los que pertenece el alumno. Además, los routes de Angular permiten la navegación entre diferentes vistas, como la página principal del alumno, su dashboard, y las secciones de cuestionarios resueltos o pendientes. Esta estructura permite que el sistema sea modular, escalable y fácil de mantener.



Anexo 66. Estructura del proyecto



Anexo 67. Estructura del proyecto (Frontend)



Anexo 68. Estructura del proyecto (Frontend)

```
"dependencies": {
  "@angular/animations": "^18.0.0",
  "@angular/cdk": "^18.2.5",
  "@angular/common": "^18.0.0",
  "@angular/compiler": "^18.0.0",
  "@angular/core": "^18.0.0",
  "@angular/forms": "^18.0.0",
  "@angular/material": "^18.2.5",
  "@angular/platform-browser": "^18.0.0",
  "@angular/platform-browser-dynamic": "^18.0.0",
  "@angular/platform-server": "^18.0.0",
  "@angular/router": "^18.0.0",
  "@angular/ssr": "^18.0.3",
  "@fortawesome/fontawesome-free": "^6.6.0",
  "@popperjs/core": "^2.11.8",
  "bootstrap": "^5.3.3",
  "bootstrap-icons": "^1.11.3",
  "chart.js": "^4.4.4",
  "express": "^4.18.2",
  "material-design-icons-iconfont": "^6.7.0",
  "rxjs": "~7.8.0",
  "sweetalert": "^2.1.2",
  "sweetalert2": "^11.14.0",
  "tslib": "^2.3.0",
  "zone.js": "~0.14.3"
},
```

Anexo 69. Librerías usadas del proyecto (Frontend)

Componentes

Los componentes de Angular que hemos creado son las piezas fundamentales para construir la interfaz de usuario de nuestra aplicación. Cada componente tiene una lógica específica y es responsable de la presentación de una parte del sistema. Los componentes interactúan entre sí a través de servicios y los datos son manejados y presentados de manera eficiente. Algunos de los componentes clave incluyen aquellos para la visualización de cuestionarios, resultados, gestión de grupos y administración de perfiles de usuario. Estos permiten que la plataforma sea interactiva y dinámica, facilitando la experiencia del usuario.

Para el registro del alumno se creó un formulario donde se añadirán los datos del estudiante, en ella se adecua el tipo de usuario del alumno, así como los validadores necesarios para que se respeten la estructura del registro.

```
this.formGroup = new FormGroup({
  nombre: new FormControl('', [nameValidator, makeRequired, lengthValidator({ min: 1, max: 30 })]),
  papellido: new FormControl('', [nameValidator, makeRequired, lengthValidator({ min: 1, max: 15 })]),
  sapellido: new FormControl('', [nameValidator, makeRequired, lengthValidator({ min: 1, max: 15 })]),
  matricula: new FormControl('', [makeRequired, lengthValidator({ min: 1, max: 15 })]),
  semestre: new FormControl('', [makeRequired]),
  tipo: new FormControl('Alumno', []),
  correo: new FormControl('', [emailValidator(), makeRequired, lengthValidator({ min: 1, max: 30 })]),
  contraseña: new FormControl('', [makeRequired, lengthValidator({ min: 8, max: 15 })]),
  confirmcontraseña: new FormControl('', [makeRequired, lengthValidator({ min: 8, max: 15 })]),
  edad: new FormControl('', [makeRequired]),
  genero: new FormControl('', [makeRequired]),
  institucion: new FormControl('', [makeRequired]),
  pais: new FormControl('', [makeRequired]),
  carrera: new FormControl('', [makeRequired])
}, { validators: confirmPassword });
```

Anexo 70. Formulario del alumno

De igual manera en el formulario se agregaron 3 subscripciones a un observable las cuales tienen la finalidad de obtener los países e instituciones del sistema y una función el cual mostrará una lista de edades para que el usuario seleccione la edad en un rango de 17 a 99 años.

```

this.catalogs.getCountries().subscribe((response: ResponseGet<Countries[]>) => {
  this.Countries = response.data;
});

this.formGroup.get('pais')?.valueChanges.subscribe(selectedCountryId => {
  this.onCountryChange(selectedCountryId);
});

this.formGroup.get('institucion')?.valueChanges.subscribe(selectedInstitutionId => {
  this.onInstitutionChange(selectedInstitutionId);
});

this.ages = Array.from({ length: 99 - 17 + 1 }, (_, i) => i + 17);
}

```

Anexo 71. Subscripciones para países e instituciones y rango de edad

El componente tiene 2 funciones importantes, la primera función “onCountryChange” tiene la finalidad de obtener las instituciones que pertenecen a un país, es decir, filtra las instituciones de un país. La segunda función “onInstitutionChange” tiene la misma funcionalidad, pero en este caso Filtra las carreras que pertenecen a la institución seleccionada.

```
Complexity is 3 Everything is cool!
onCountryChange(IDPais: string) {
  if (IDPais) {
    this.catalogs.getInstitutionsId(IDPais).subscribe((response: ResponseGet<Institutions[]>) => {
      this.FilteredInstitutions = response.data;
    });
  } else {
    this.FilteredInstitutions = [];
  }
}

Complexity is 3 Everything is cool!
onInstitutionChange(InstitucionID: string) {
  if (InstitucionID) {
    this.catalogs.getDegreesId(InstitucionID).subscribe((response: ResponseGet<Degrees[]>) => {
      this.FilteredDegrees = response.data;
    });
  } else {
    this.FilteredDegrees = [];
  }
}
```

Anexo 72. Funciones para obtener instituciones y carreras filtradas

La función “onSubmit” sirve para mandar los campos que han sido llenados por el usuario y a partir del servicio registerStudent almacena los datos del nuevo usuario y despliega un mensaje de que el usuario se ha creado correctamente o muestra un error si el usuario no se pudo crear.

```
Complexity is 7 It's time to do something...
onSubmit() {
  if (this.formGroup.invalid) {
    this.formGroup.markAllAsTouched();
  } else {
    const createStudent: createStudent = {
      Nombre: this.formGroup.value.nombre,
      Primer_apellido: this.formGroup.value.papellido,
      Segundo_apellido: this.formGroup.value.sapellido,
      Matricula: this.formGroup.value.matricula,
      Semestre: this.formGroup.value.semestre,
      Tipo: this.formGroup.value.tipo,
      Correo: this.formGroup.value.correo,
      Password: this.formGroup.value.contraseña,
      edad: this.formGroup.value.edad,
      genero: this.formGroup.value.genero,
      InstitucionID: this.formGroup.value.institucion,
      IDPais: this.formGroup.value.pais,
      CarreraID: this.formGroup.value.carrera
    };
    this.userService.registerStudent(createStudent).subscribe(
      (response) => {
        Swal.fire({
          title: '¡Éxito!',
          text: 'Estudiante creado correctamente',
          icon: 'success',
          confirmButtonText: 'Aceptar'
        });
      },
      (error) => {
        if (error.status === 422 && error.error.message) {
          ServerErrorValidator.setServerErrors(error.error.message, this.formGroup);
        } else {
          Swal.fire({
            title: 'Error',
            text: 'Ocurrió un error al crear el estudiante.',
            icon: 'error',
            confirmButtonText: 'Aceptar'
          }).then(() => {
            this.goBack();
          });
        }
      }
    );
  }
}
```

Anexo 73. Función onSubmit

```

1 <div class="container">
2   
3   <h2 class="form-title">Registrar estudiante</h2>
4
5   <form [formGroup]="formGroup" (ngSubmit)="onSubmit()">
6
7     <div class="form-group">
8       <text-input label="Nombres" [controller]="formGroup.controls['nombre']" type="text"></text-input>
9     </div>
10
11     <div class="form-group form-group-inline">
12       <text-input label="Primer Apellido" [controller]="formGroup.controls['papellido']" type="text"></text-input>
13       <text-input label="Segundo Apellido" [controller]="formGroup.controls['sapellido']" type="text"></text-input>
14     </div>
15
16     <div class="form-group form-group-inline">
17       <div class="form-group">
18         <label for="genero">Género</label>
19         <select class="form-select" id="genero" formControlName="genero">
20           <option value="">Seleccione un género</option>
21           <option value="Masculino">Masculino</option>
22           <option value="Femenino">Femenino</option>
23           <option value="Otro">Otro</option>
24         </select>
25         <div *ngIf="formGroup.get('genero')?.invalid && formGroup.get('genero')?.touched">
26           <span class="error-message">Selecciona un género válido</span>
27         </div>
28       </div>
29
30       <div class="form-group">
31         <label for="edad">Edad</label>
32         <select class="form-select" id="edad" formControlName="edad">
33           <option value="">Seleccione una edad</option>
34           <option *ngFor="let age of ages" [value]="age">{{ age }}</option>
35         </select>
36         <div *ngIf="formGroup.controls['edad'].invalid && formGroup.controls['edad'].touched">
37           <span class="error-message">Selecciona una edad válida</span>
38         </div>
39       </div>
40     </div>

```

Anexo 74. Código para la vista del registro 1

```

42   <div class="form-group-inline">
43     <div class="form-group">
44       <label for="pais">País</label>
45       <select class="form-select" id="pais" formControlName="pais">
46         <option value="">Seleccione un país</option>
47         <option *ngFor="let IDPais of Countries" [value]="IDPais.IDPais">
48           {{ IDPais.Nombre }}
49         </option>
50       </select>
51       <div *ngIf="formGroup.get('pais')?.invalid && formGroup.get('pais')?.touched">
52         <span class="error-message">Selecciona un país válido</span>
53       </div>
54     </div>
55
56     <div class="form-group" *ngIf="formGroup.get('pais')?.value">
57       <label for="institucion">Institución</label>
58       <select class="form-select" id="institucion" formControlName="institucion">
59         <option value="">Seleccione una institución</option>
60         <option *ngFor="let institucion of FilteredInstitutions" [value]="institucion.InstitucionID">
61           {{ institucion.Nombre }}
62         </option>
63       </select>
64       <div *ngIf="formGroup.get('institucion')?.invalid && formGroup.get('institucion')?.touched">
65         <span class="error-message">Selecciona una institución válida</span>
66       </div>
67     </div>
68   </div>
69
70   <div class="form-group">
71     <div class="form-group">
72       <label for="tipo">Tipo de usuario</label>
73       <select class="form-select" id="tipo" formControlName="tipo" [disabled]="true">
74         <option value="Alumno" selected>Alumno</option>
75       </select>
76     </div>
77   </div>

```

Anexo 75. Código para la vista del registro 2

```

79     <div class="form-group-inline">
80         <div class="form-group">
81             <div class="form-group">
82                 <label for="tipo">Semestre</label>
83                 <select class="form-select" id="semestre" formControlName="semestre">
84                     <option value="">Semestre actual</option>
85                     <option value="2º" selected>2º</option>
86                     <option value="6º" selected>6º</option>
87                     <option value="8º" selected>8º</option>
88                 </select>
89                 <div *ngIf="formGroup.get('semestre')?.invalid && formGroup.get('semestre')?.touched">
90                     <span class="error-message">Seleccione un semestre</span>
91                 </div>
92             </div>
93         </div>
94
95         <text-input label="Matricula" [controller]="formGroup.controls['matricula']" type="text"></text-input>
96         <div *ngIf="formGroup.get('matricula')?.errors?.['serverError']">
97             <span class="error-message">{{ formGroup.get('matricula')?.errors?.['serverError'] }}</span>
98         </div>
99     </div>
100
101     <div class="form-group" *ngIf="formGroup.get('institucion')?.value">
102         <label for="carrera">Carrera</label>
103         <select class="form-select" id="carrera" formControlName="carrera">
104             <option value="">Seleccione una carrera</option>
105             <option *ngFor="let carrera of FilteredDegrees" [value]="carrera.CarreraID">
106                 {{ carrera.Nombre }}
107             </option>
108         </select>
109         <div *ngIf="formGroup.get('carrera')?.invalid && formGroup.get('carrera')?.touched">
110             <span class="error-message">Seleccione una carrera válida</span>
111         </div>
112     </div>

```

Anexo 76. Código para la vista del registro 3

```

116     <div class="form-group">
117         <text-input label="Correo Electrónico" [controller]="formGroup.controls['correo']" type="text"></text-input>
118         <div *ngIf="formGroup.get('correo')?.errors?.['serverError']">
119             <span class="error-message">{{ formGroup.get('correo')?.errors?.['serverError'] }}</span>
120         </div>
121     </div>
122
123     <div class="form-group">
124         <text-input label="Contraseña" [controller]="formGroup.controls['contraseña']" type="password"></text-input>
125     </div>
126
127     <div class="form-group">
128         <text-input label="Confirmar contraseña" [controller]="formGroup.controls['confirmcontraseña']" type="password"></text-input>
129         <div *ngIf="formGroup.errors?.['passwordMismatch'] && formGroup.get('confirmcontraseña')?.touched">
130             <span class="error-message">{{ formGroup.errors?.['passwordMismatch'].message }}</span>
131         </div>
132     </div>
133
134     <button type="button" class="btn btn-back" [routerLink]="['/home']">Regresar</button>
135     <button type="submit" class="btn btn-register">Registrarse</button>
136
137 </form>
138 </div>

```

Anexo 77. Código para la vista del registro 4

En la vista de la página principal está compuesta por 6 contenedores el cual contiene: header, componente principal, carrusel de imágenes, objetivo del cuestionario, aclaración y footer.

```

14'      <!-- Opciones del menú -->
15      <div class="collapse navbar-collapse nav-underline" id="navbarNav">
16        <ul class="navbar-nav mx-auto">
17          <hr class="text-black-50">
18          <li class="nav-item">
19            <a class="nav-link text-black" [routerLink]="['/home']">Home</a>
20          </li>
21          <li class="nav-item dropdown">
22            <a class="nav-link text-black dropdown-toggle" [routerLink]="['/quien_somos']"
23              id="navbarDarkDropdownMenuLink" role="button" data-bs-toggle="dropdown" aria-expanded="false">
24              Quienes somos
25            </a>
26          </li>
27          <li class="nav-item">
28            <a class="nav-link text-black" [routerLink]="['/locacion']">Contacto</a>
29          </li>
30          <li class="nav-item">
31            <a class="nav-link text-black" [routerLink]="['/instrucciones']">Cuestionario</a>
32          </li>
33        </ul>
34      </div>
35      <!-- Botones para el iniciar sesion y registrar -->
36      <hr class="text-white-50 bg-warning">
37      <div class="d-grid gap-2 d-md-flex justify-content-md-end">
38        <button type="button" class="btn btn-outline-signup me-md-2" [routerLink]="['/registro']">SIGN UP</button>
39        <button type="button" class="btn btn-login ml-2" [routerLink]="['/login']">LOG IN</button>
40      </div>
41    </div>
42  </nav>

```

Anexo 78. Opciones del menú del header

```

1  <body class="contenedorprincipal">
2  <!-- Contenedor principal con la imagen de fondo -->
3  <div class="container-fluid fondo-imagen d-flex align-items-center alto">
4    <div class="fondo-degradado ml-auto text-center p-4">
5      <h1 class="titulo">Autoevaluación de las Características Emprendedoras Personales</h1>
6      <button type="button" class="btn btn-primary btn-lg mt-4" [routerLink]="['/instrucciones']">¡CONTESTA AHORA!</button>
7    </div>
8  </div>
9  <div class="carrusel">
10    <app-carrusel></app-carrusel>
11  </div>
12
13  <div class="contenedor1">
14    <app-nota></app-nota>
15  </div>
16
17  <div class="contenedor2">
18    <app-nota2></app-nota2>
19  </div>
20 </body>

```

Anexo 79. Contenedor principal


```

Go to component | Fernanda-Chable, hace 4 meses | 1 author (Fernanda-Chable)
1 <div class="container-fluid fondo">
2   <div class="content">
3     <div class="text">
4       <h1>¿Cuál es el objetivo del cuestionario?</h1>
5       <p>Su objetivo principal es que la persona misma pueda analizar y construir
6         su propio perfil acerca de estas características.
7         La identificación de sus debilidades y sus fuerzas le da la posibilidad de
8         reforzar o aprovechar mejor sus propias condiciones.
9       </p>
10    </div>
11    <div class="images">
12      
13      
14    </div>
15  </div>
16 </div>

```

Anexo 80. Objetivo del cuestionario

```

<footer class="bg-dark text-white py-5 footer-with-top-margin">
  <div class="container">
    <div class="row">
      <!-- Logo y Proyecto -->
      <div class="col-md-4 mb-2 d-flex flex-column justify-content-between">
        <div>
          <a href="#" class="text-reset text-uppercase d-flex align-items-center logo-link mb-3">
            
            <span class="logo-text"></span>
          </a>
          <h3>Proyecto Educativo</h3>
          <p>Este proyecto es una colaboración entre maestros de
            diversas instituciones para que los alumnos puedan identificar sus debilidades
            y fuerzas en su perfil emprendedor. Nuestro objetivo es que la persona misma
            pueda analizar y construir su propio perfil acerca de estas características.
            La identificación de sus debilidades y sus fuerzas le
            da la posibilidad de reforzar o aprovechar
            mejor sus propias condiciones. </p>
        </div>
      </div>
      <!-- Contacto e Instituciones -->
      <div class="col-md-5 mb-2 d-flex flex-column justify-content-between">
        <div>
          <h3>Créditos</h3>
          <p>Este proyecto ha sido desarrollado con el esfuerzo
            y la dedicación de varias personas.
            Los maestros de diversas instituciones educativas
            han liderado la investigación y el diseño del proyecto,
            mientras que dos alumnos que están por egresar del Instituto Tecnológico Superior
            de Calkini han sido responsables del desarrollo de la página web.
            Agradecemos a todos los colaboradores y participantes en
            este proyecto.</p>
        </div>
        <div>
          <h3>Instituciones Participantes</h3>
          <ul class="list-unstyled">
            <li><a href="https://www.conkal.tecnm.mx/" class="text-reset">Instituto Tecnológico de Conkal</a></li>
            <li><a href="https://hopelchen.tecnm.mx/portal/" class="text-reset">Instituto Tecnológico Superior de Hopelchén</a></li>
            <li><a href="https://itescam.edu.mx/portal/" class="text-reset">Instituto Tecnológico Superior de Calkini</a></li>
          </ul>
        </div>
      </div>
    </div>
  </div>

```

Anexo 81. Código del Footer 1

```

46 <!-- Redes sociales y Créditos -->
47 <div class="col-md-3 mb-2 d-flex flex-column justify-content-between">
48   <div class="social-icons mb-3">
49     <h3>Síguenos en nuestras Redes Sociales</h3>
50     <hr class="social-line">
51     <ul class="list-unstyled d-flex justify-content-between">
52       <!-- Facebook -->
53       <li><a href="https://www.facebook.com/?locale=es_LA" class="facebook">
54         <i class="bi bi-facebook fs-2"></i>
55       </a></li>
56       <!-- Twitter -->
57       <li><a href="https://twitter.com/?lang=es" class="x">
58         <i class="bi bi-twitter fs-2"></i>
59       </a></li>
60       <!-- Instagram -->
61       <li><a href="https://www.instagram.com" class="instagram">
62         <i class="bi bi-instagram fs-2"></i>
63       </a></li>
64       <!-- WhatsApp -->
65       <li><a href="https://www.whatsapp.com" class="whatsapp">
66         <i class="bi bi-whatsapp fs-2"></i>
67       </a></li>
68     </ul>
69   </div>
70   <div>
71     <h3>Contacto</h3>
72     <ul class="list-unstyled">
73       <li><strong>Email:</strong> </li>
74       <li><strong>Teléfono:</strong> +9961026754</li>
75     </ul>
76   </div>
77 </div>
78 </div>

```

Anexo 82. Código del Footer parte 2

```

79 <div class="row">
80   <div class="col text-center">
81     <ul class="list-unstyled d-inline-block">
82       <li><a href="/terminos-de-uso" class="text-reset">Términos de Uso</a></li>
83       <li><a href="/politica-de-privacidad" class="text-reset">Política de Privacidad</a></li>
84     </ul>
85     <p class="mt-2">&copy; 2024 Proyecto Educativo. Todos los derechos reservados.</p>
86   </div>
87 </div>
88 </div>
89 </footer>
90 </div>

```

Anexo 83. Código del Footer parte 3

Para el apartado de localización se contiene la información del tecnológico la estructura para una institución se repite de manera similar para la información de las otras dos instituciones.

```

2 <div class="title-container">
3   
4   <h1 class="titulo">DONDE ENCONTRARNOS</h1>
5 </div>
6 <br>
7 <!-- Contenedor del instituto de calkini-->
8 <div class="content-container">
9   <div class="title-container">
10    
11    <h1 class="centered-title">Instituto Tecnológico Superior de Calkiní en el estado de campeche</h1>
12  </div>
13  <br>
14  <div class="content-below">
15    <p class="text-left">
16      Dirección: Avenida AH-Canun, Sin Número, San Felipe, 24900 Calkiní, Camp.<br>
17      Teléfono: 996 813 4870
18    <br>
19    <a href="https://www.itescam.edu.mx/" class="text-reset">Sitio web: https://www.itescam.edu.mx</a>
20  </p>
21  <div class="map-right">
22    <iframe src="https://www.google.com/maps/embed?pb=!1m14!1m8!1m3!1d14254.95661178538!2d-90.0442127610134!3d20.34504712022303!3m2!1s"
23      width="100%" height="100%" style="border:0;" allowfullscreen="" loading="lazy"></iframe>
24  </div>
25 </div>
26 </div>

```

Anexo 84. Código de contactos

```
<div class="title-container">
  
  <h1 class="titulo">¿QUIENES SOMOS?</h1>
</div>
<div class="container">
  <h1 class="titulo">NUESTRA HISTORIA</h1>
</div>
```

Anexo 85. Código de "Quienes somos"

Para el login de usuarios se creó el formulario el cual admite correo o matrícula y la contraseña. De igual manera una función la cual a partir del AuthService se obtiene el tipo de usuario y redirige al usuario a su respectivo Dashboard con un switch.

```

22 login() {
23   this.authService.login(this.FormGroup).subscribe(
      Complexity is 10 It's time to do something...
24     (response) => {
25       console.log('Respuesta del API:', response);
26       localStorage.setItem('access_token', response.access_token);
27       localStorage.setItem('user_id', response.usuario.UsuarioID.toString());
28     }
29     // Obtenemos el tipo de usuario
30     const tipoUsuario = response.usuario.Tipo;
31
32     // Redirigimos al dashboard correspondiente según el tipo de usuario
33     switch (tipoUsuario) {
34       case 'Alumno':
35         this.router.navigate(['/alumno']);
36         break;
37       case 'Maestro':
38         this.router.navigate(['/maestro']);
39         break;
40       case 'Admin':
41         this.router.navigate(['/administrador']);
42         break;
43       default:
44         this.router.navigate(['/home']);
45         break;
46     }
47   },
48   (error) => {
49     console.error('Login failed', error);
50   }
51 );
52 }
53 }
54 }

```

Anexo 86. Función de Login

```

1 <link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
2 <div class="login-container">
3   <div class="login-box">
4     <span class="material-icons">account_circle</span>
5     <h1>Login</h1>
6
7     <form (submit)="login()">
8       <label for="Correo">Correo electrónico o Matrícula:</label>
9       <input type="text" [(ngModel)]="FormGroup.Correo" name="Correo" required />
10
11       <label for="Password">Contraseña:</label>
12       <input type="password" [(ngModel)]="FormGroup.Password" name="Password" required />
13
14       <div class="button-group">
15         <button type="button" class="btn regresar-btn">Regresar</button>
16         <button type="submit" class="btn login-btn">Iniciar sesión</button>
17       </div>
18
19       <a href="#" class="forgot-password">¿Olvidaste tu contraseña?</a>
20     </form>
21   </div>
22 </div>

```

Anexo 87. Código de la vista del login

Para el Dashboard del estudiante tiene su propio menú el cual es donde podrá navegar el alumno. Donde al ingresar muestra el nombre, apellidos del alumno, así como su correo.

```

<div *ngIf="isLoggedIn" class="dashboard">
  <div class="sidebar">
    <div class="user-info">
      
      <p *ngIf="student">{{ student.Nombre }} {{ student.Primer_apellido }} {{ student.Segundo_apellido }}</p>
      <p *ngIf="student">{{ student.Correo }}</p>
    </div>
    <div class="menu">
      <a [routerLink]="['/alumno/dash-estu']">
        <mat-icon>home</mat-icon> Home
      </a>
      <a [routerLink]="['/alumno/gruops']">
        <mat-icon>groups</mat-icon> Grupos
      </a>
      <a [routerLink]="['/alumno/generales']">
        <mat-icon>quiz</mat-icon> Cuestionarios resueltos
      </a>
      <a (click)="logout()" *ngIf="isLoggedIn">
        <mat-icon>logout</mat-icon> Cerrar Sesión
      </a>
    </div>
  </div>
  <div class="main-content">
    <div class="content-container">
      <router-outlet></router-outlet>
    </div>
  </div>
</div>

```

Anexo 88. Código de la vista del menú

```

35   loadStudentData(): void {
36     const studentId = this.authService.getUserId();
37     console.log('ID del estudiante:', studentId);
38
39     if (studentId) {
40       this.userService.getStudent(studentId).subscribe(
41         response => {
42           console.log('Respuesta del API:', response);
43           if (response.data) {
44             this.student = response.data;
45             console.log('Datos del estudiante:', this.student);
46           } else {
47             console.error('No se encontraron datos del estudiante');
48           }
49         },
50         error => {
51           console.error('Error al cargar datos del estudiante', error);
52         }
53       );
54     } else {
55       console.error('No se encontró un ID de usuario autenticado');
56     }
57   }

```

Anexo 89. Función para cargar información del alumno del menú.

Para el Dashboard del alumno se agregó una bienvenida con la fecha del día el cual inició sesión, también incluye un menú el cual puede observar la cantidad de grupos, resultados, cuestionarios resueltos y cuestionarios pendientes.

```

1 <div class="main-content">
2   <header class="header">
3     <div class="header-box">
4       <h1>Bienvenido, {{ currentDate }}</h1>
5     </div>
6     <div class="search-and-button">
7       <input type="search" placeholder="Search">
8       <button class="entrar-grupo" (click)="openEntrarGrupo()">Entrar al grupo</button>
9     </div>
10  </header>
11
12  <section class="home">
13    <div class="home-box">
14      <h1><mat-icon>home</mat-icon> HOME</h1>
15    </div>
16  </section>
17
18  <div class="buttons">
19    <button class="button blue" [routerLink]="['/alumno/grupos']">
20      <mat-icon>group</mat-icon> Grupos <span class="counter">{{ totalGrupos }}</span>
21    </button>
22    <button class="button orange" [routerLink]="['/alumno/Resultados_generales']">
23      <mat-icon>bar_chart</mat-icon> Resultados <span class="counter">{{ totalResultados }}</span>
24    </button>
25    <button class="button green" [routerLink]="['/alumno/generales']">
26      <mat-icon>check_circle</mat-icon> Cuestionarios resueltos <span class="counter">{{ totalCuestionariosResueltos }}</span>
27    </button>
28    <button class="button pink" [routerLink]="['/alumno/pendiente']">
29      <mat-icon>schedule</mat-icon> Cuestionarios pendientes <span class="counter">{{ totalCuestionariosPendientes }}</span>
30    </button>
31  </div>
32 </div>
33
34 <app-enter-group *ngIf="showEnterGroupPopup" (close)="closePopup()"></app-enter-group>
35

```

Anexo 90. Código visto del dashboard.

Se realizaron funciones para obtener la fecha de bienvenida y contadores del total de grupos, total de resultados, cuestionarios resueltos, cuestionarios pendientes.

Al ingresar al Dashboard el componente obtiene el id del usuario y la función para obtener el día, de igual manera inicializa las funciones para obtener los contadores del total de grupos, total de cuestionarios resueltos, total de cuestionarios pendientes y total de resultados.

```
39  ngOnInit(): void {
40      this.usuarioID = this.authService.getUserId();
41      this.currentDate = this.getFormattedDate();
42      this.getTotalGrupos();
43      this.getTotalResultados();
44      this.getTotalCuestionariosResueltos();
45      this.getTotalCuestionariosPendientes();
46  }
47
48  // Función para formatear la fecha
49  getFormattedDate(): string {
50      const today = new Date();
51      const daysOfWeek = ['Domingo', 'Lunes', 'Martes', 'Miércoles', 'Jueves', 'Viernes', 'Sábado'];
52      const monthsOfYear = ['Enero', 'Febrero', 'Marzo', 'Abril', 'Mayo', 'Junio', 'Julio', 'Agosto',
53          'Septiembre', 'Octubre', 'Noviembre', 'Diciembre'];
54
55      const dayOfWeek = daysOfWeek[today.getDay()];
56      const dayOfMonth = today.getDate();
57      const month = monthsOfYear[today.getMonth()];
58      const year = today.getFullYear();
59
60      return `${dayOfWeek} ${dayOfMonth} de ${month} del ${year}`;
61  }
62
63  Complexity is 5 Everything is cool!
64  getTotalGrupos(): void {
65      if (this.usuarioID !== null) {
66          this.alumnoGruposService.getAlumnoGrupos(this.usuarioID).subscribe(
67              (response) => {
68                  if (response.data) {
69                      this.totalGrupos = response.data.length;
70                  } else {
71                      this.totalGrupos = 0;
72                  }
73              },
74              (error) => {
75                  console.error('Error al obtener grupos:', error);
76              }
77          );
78      }
79  }
```

Anexo 91. Inicializador de funciones, función “formattedDate” y “getTotalGrupos”.

```

80   getTotalResultados(): void {
81     if (this.usuarioID !== null) {
82       this.solvedQuizzesService.getCuestionariosResueltosPorAlumno(this.usuarioID).subscribe(
83         Complexity is 3 Everything is cool!
84         (response) => {
85           if (response.data && response.data.length) {
86             this.totalResultados = response.data.length;
87           } else {
88             this.totalResultados = 0;
89           }
90         },
91         (error) => {
92           console.error('Error al obtener resultados:', error);
93         }
94       );
95     }
96     Complexity is 4 Everything is cool!
97   }
98   getTotalCuestionariosResueltos(): void {
99     if (this.usuarioID !== null) {
100       this.solvedQuizzesService.getCuestionariosResueltosPorAlumno(this.usuarioID).subscribe(
101         (response) => {
102           this.totalCuestionariosResueltos = response.data.length;
103         },
104         (error) => {
105           console.error('Error al obtener cuestionarios resueltos:', error);
106         }
107       );
108     }
109   }

```

Anexo 92. Funciones “getTotalResultados” y “getTotalCuestionariosResueltos”

```

108   getTotalCuestionariosPendientes(): void {
109     if (this.usuarioID !== null) {
110       this.alumnoGruposService.getAlumnoGrupos(this.usuarioID).subscribe(
111         Complexity is 6 It's time to do something...
112         (response) => {
113           if (response.data && response.data.length > 0) {
114             let totalPendientes = 0;
115             Complexity is 3 Everything is cool!
116             response.data.forEach((grupo) => {
117               const grupoId = grupo.GrupoID;
118               console.log('Grupo ID:', grupoId);
119               this.solvedQuizzesService.obtenerCuestionariosPendientes(this.usuarioID!, grupoId).subscribe(
120                 (cuestionariosResponse) => {
121                   totalPendientes += cuestionariosResponse.data.length;
122                   this.totalCuestionariosPendientes = totalPendientes;
123                 },
124                 (error) => {
125                   console.error('Error al obtener cuestionarios pendientes para el grupo', grupoId, ':', error);
126                 }
127               );
128             });
129           } else {
130             this.totalCuestionariosPendientes = 0;
131           }
132         },
133         (error) => {
134           console.error('Error al obtener grupos:', error);
135           this.totalCuestionariosPendientes = 0;
136         }
137       );
138     } else {
139       console.error('usuarioID es null');
140       this.totalCuestionariosPendientes = 0;
141     }
142   }

```

Anexo 93. Función “GetTotalCuestionariosPendientes”

Para ingresar a un grupo mediante una pop-up donde se ingresa el código de acceso, por lo que se creó una función “buscarGrupo” el cual a partir de buscar por código mediante el servicio “getGrupoByCodigo” y otra función “unirseGrupo” el cual permite al usuario ingresarlo al grupo relacionándolo con su id. También se agregó una función “cancelar” el cual como su nombre lo indica, cancela el proceso para ingresar a un grupo.

```
28     buscarGrupo() {
29         const codigoAcceso = this.form.get('codigo')?.value;
30         if (this.form.invalid) {
31             return;
32         }
33         this.loading = true;
34         const codigo = this.form.value.codigo_acceso;
35         this.grupoService.getGrupoByCodigo(codigo).subscribe(
36             (response) => {
37                 this.grupoInfo = response.data;
38                 this.loading = false;
39             },
40             (error) => {
41                 this.error = error.error.message;
42                 this.loading = false;
43                 this.cancelar();
44             }
45         );
46     }
47
48     Complexity is 3 Everything is cool!
49     unirseGrupo(usuarioId: number) {
50         const codigo = this.form.value.codigo_acceso;
51         this.grupoService.joinGrupo(usuarioId, codigo).subscribe(
52             (response) => {
53                 Swal.fire('Exito', 'Te has unido al grupo correctamente', 'success');
54             },
55             (error) => {
56                 Swal.fire('Error', error.error.message, 'error');
57             }
58         );
59     }
60     cancelar() {
61         this.form.reset();
62         this.grupoInfo = null;
63         this.error = '';
64         this.close.emit();
65     }
66 }
```

Anexo 94. Funciones “buscarGrupo”, “unirseGrupo” y “cancelar”


```

<div class="overlay">
  <div class="pop-up">
    <button class="close-button" (click)="cancelar()"><mat-icon>close</mat-icon></button>
    <h2>Busca el Grupo</h2>

    <div class="input-group">
      <form [formGroup]="form" (ngSubmit)="buscarGrupo()">
        <label for="codigo_acceso">Código de Acceso:</label>
        <input id="codigo_acceso" formControlName="codigo_acceso" type="text" />
        <button type="submit" [disabled]="loading">Buscar Grupo</button>
      </form>
    </div>

    <div *ngIf="grupoInfo" class="grupo-container">
      <div class="grupo-info">
        <h3>GRUPO: {{ grupoInfo.Nombre }}</h3>
        <p><strong>Institución:</strong> {{ grupoInfo.carrera.institucion.Nombre }}</p>
        <p><strong>Semestre:</strong> {{ grupoInfo.semestre }}</p>
        <p><strong>Carrera:</strong> {{ grupoInfo.carrera.Nombre }}</p>
      </div>
      <div class="grupo-actions">
        <button (click)="unirseGrupo(1)">Unirse</button>
        <button (click)="cancelar()">Cancelar</button>
      </div>
    </div>
    <div *ngIf="error">{{ error }}</div>
  </div>
</div>

```

Anexo 95. Código de la vista de la ventana buscar grupo

Para la sección grupos se codificó la vista para mostrar la lista de grupos en las que está escrito el alumno estas tarjetas contiene información del grupo como el nombre de institución, carrera, semestre y nombre del grupo, también contiene un botón para ingresar a un grupo. La información del grupo se mostrará en una tarjeta con una función para visualizar los cuestionarios resueltos de ese grupo.

```

1  <section class="home">
2    <div class="home-box">
3      <h1><mat-icon>home</mat-icon> MIS GRUPOS</h1>
4    </div>
5  </section>
6
7  <div class="groups-header">
8    <button class="enter-group-btn" (click)="openEntrarGrupo()">Entrar al Grupo</button>
9  </div>
10
11 <div *ngIf="alumno">
12   <h2>{{ alumno.Nombre }} {{ alumno.Primer_apellido }} {{ alumno.Segundo_apellido }}</h2>
13 </div>
14
15 <div class="groups-container">
16   <div *ngFor="let group of groups; let i = index" (click)="enterGroup(group)" style="cursor: pointer;">
17     <div class="group-card">
18       <div class="icon-container" [ngStyle]="{'background': getRandomGradient()}">
19         <i class="material-icons">group</i>
20       </div>
21       <div class="info-container">
22         <p>{{ group.carrera?.institucion?.Nombre }}</p>
23         <p>{{ group.carrera.Nombre }}</p>
24         <p>Semestre: {{ group.semestre }} &nbsp; Grupo: {{ group.Nombre }}</p>
25       </div>
26     </div>
27   </div>
28 </div>
29
30 <div *ngIf="!groups || groups.length === 0">No hay grupos disponibles.</div>
31
32 <app-enter-group *ngIf="showEnterGroupPopup" (close)="closePopup()"></app-enter-group>
33
34

```

Anexo 96. Código de la vista de grupos.

Para el funcionamiento de esta vista se crearon ciertas funciones. Se añadió un inicializador para obtener el id del usuario para la autenticación. Se creó la función “getAlumnogrupos” el cual obtiene a partir del servicio “alumnoGruposService” los grupos a los que le pertenece el alumno. Otra función que se agregó fue la de “enterGroup” la cual redirige al alumno a la vista de los cuestionarios resueltos del grupo.

```

29 ngOnInit() {
30   const usuarioID = this.authService.getUserId();
31   if (usuarioID) {
32     this.getAlumnoGrupos(usuarioID);
33   } else {
34     console.error('No se encontró el usuario autenticado');
35   }
36   setTimeout(() => {
37     this.getRandomGradient();
38   });
39 }
40
41 getAlumnoGrupos(usuarioID: number) {
42   this.alumnoGruposService.getAlumnoGrupos(usuarioID).subscribe((response: any) => {
43     this.groups = response.data;
44     console.log('Grupos obtenidos:', this.groups);
45   });
46 }
47
48 enterGroup(group: { GrupoID: number }) {
49   console.log('Entering group:', group);
50   const groupId = group.GrupoID;
51   this.router.navigate(['/alumno/cuestionarios-resueltos'], { queryParams: { groupId } });
52 }
53

```

Anexo 97. Inicializador, Funciones “getAlumnos” y “enterGroup”.

Ahora para la vista de cuestionarios resueltos por grupo, se creó la vista el cual muestra la información del grupo tales como: la institución, carrera, semestre y grupo. También una lista de los cuestionarios resueltos por el alumno mostrando la información de la versión del cuestionario y la fecha la cual se resolvió. Si no existen los cuestionarios mostrará “No hay cuestionarios resueltos disponibles”.

```

2 <div *ngIf="grupo && grupo.Nombre && alumno" class="grupo-info">
3   <mat-icon class="grupo-icon">group</mat-icon>
4   <div class="grupo-text">
5     <p *ngIf="grupo.carrera?.institucion?.Nombre">{{ grupo.carrera.institucion.Nombre }}</p>
6     <p *ngIf="grupo.carrera?.Nombre">{{ grupo.carrera.Nombre }}</p>
7     <p>Semestre: {{ grupo.semestre }} &nbsp; Grupo: {{ grupo.Nombre }}</p>
8   </div>
9 </div>
10 <section class="home">
11   <div class="home-box">
12     <h1><mat-icon>check_circle</mat-icon> CUESTIONARIOS RESUELTOS</h1>
13   </div>
14 </section>
15
16 <div class="groups-container">
17   <div *ngFor="let quiz of solvedQuizzes">
18     <div class="quiz-card" (click)="goToQuizResults(quiz.CuestionarioID)">
19       <div class="icon-container" [ngStyle]="{'background': getRandomGradient()}">
20         <i class="material-icons">assignment_turned_in</i>
21       </div>
22       <div class="info-container">
23         <p>{{ quiz.Nombre }}</p>
24         <p>Fecha resuelto: {{ quiz.FechaResuelto | date:'dd/MM/yyyy' }}</p>
25       </div>
26     </div>
27   </div>
28 </div>
29
30 <div *ngIf="!solvedQuizzes.length">No hay cuestionarios resueltos disponibles.</div>

```

Anexo 98. Código de la vista de cuestionarios resueltos.

Para la funcionalidad del cuestionario para que pueda obtener los cuestionarios resueltos del alumno, se inicializa el id del usuario y el id del grupo para que a partir del servicio “alumnoGruposService” y “quizService” obtenga los cuestionarios que ha resuelto el alumno del grupo en específico.

```
ngOnInit() {
  const usuarioID = Number(localStorage.getItem('user_id'));
  const groupId = this.route.snapshot.queryParamMap.get('groupId');
  console.log('GroupId recibido:', groupId);

  if (!groupId) {
    console.error('No se encontró el groupId en los parámetros');
    return;
  }
  const groupIdNumber = parseInt(groupId, 10);
  if (isNaN(groupIdNumber)) {
    console.error('El groupId no es un número válido:', groupId);
    return;
  }
  console.log('groupId convertido:', groupIdNumber);
  this.alumnoGruposService.getAlumnoGrupos(usuarioID).subscribe(
    (response) => {
      console.log('Respuesta de grupos:', response);
      const grupos = response.data;
      if (grupos && grupos.length > 0) {
        this.grupo = grupos.find(grupo => grupo.GrupoID === groupIdNumber);
        console.log('Grupo encontrado:', this.grupo);
        if (this.grupo) {
          this.quizService.obtenerCuestionariosResueltos(usuarioID, this.grupo.GrupoID).subscribe(
            (quizzes) => {
              console.log('Cuestionarios resueltos:', quizzes);
              this.solvedQuizzes = quizzes.data;
            },
            (error) => {
              console.error('Error al obtener cuestionarios resueltos:', error);
            }
          );
        } else {
          console.warn('No se encontró el grupo con el groupId proporcionado');
        }
      } else {
        console.warn('No se encontraron grupos para este alumno');
      }
    }
  );
}
```

Anexo 99. Inicializador de cuestionarios resueltos.

Para los cuestionarios pendientes se creó la vista con el mismo formato de tarjetas, en este caso contiene el nombre del cuestionario, descripción, fecha de publicación y fecha de cierre.

```
<div class="cuestionarios-container">
  <h2>Cuestionarios Pendientes</h2>
  <div class="quiz-container">
    <div *ngFor="let cuestionario of cuestionariosPendientes" >
      <div class="quiz-card" (click)="irACuestionario(cuestionario.CuestionarioID, cuestionario.GrupoID)" >
        <div class="icon-container" [ngStyle]="{'background': getRandomGradient()}">
          <i class="material-icons">pending_actions</i>
        </div>
        <div class="cuestionario-card">
          <p>{{ cuestionario.Nombre }}</p>
          <p>Descripción: {{ cuestionario.Descripcion }}</p>
          <p>Fecha de publicacion: {{cuestionario.FechaPublicacion}}</p>
          <p>Fecha de cierre: {{cuestionario.FechaCierre}}</p>
        </div>
      </div>
    </div>
  </div>
</div>
```

Anexo 100. Código de la vista del apartado cuestionarios pendientes.

Para la funcionalidad de este apartado, similar al componente anterior se inicializa el id del usuario para cargar los grupos del alumno para posteriormente cargar los cuestionarios que aun no han sido resueltos con las funciones “cargarGruposDeAlumno” y “cargarCuestionariosPendientes”. También se agregó una función para redirigir al cuestionario pendiente.

```
ngOnInit(): void {
  this.usuarioId = this.authService.getUserId();
  if (this.usuarioId !== null) {
    this.cargarGruposDelAlumno();
  } else {
    console.error('No se encontró el ID del usuario logueado.');
```

Complexity is 6 It's time to do something...

```
  }
  cargarGruposDelAlumno(): void {
    if (this.usuarioId !== null) {
      this.alumnoGruposService.getAlumnoGrupos(this.usuarioId).subscribe(
        Complexity is 3 Everything is cool!
        response => {
          if (response && response.data) {
            this.gruposDelAlumno = response.data;
            this.cargarCuestionariosPendientes();
          } else {
            console.log('No se encontraron grupos para el alumno:', response);
          }
        },
        error => {
          console.error('Error al obtener grupos del alumno:', error);
        }
      );
    }
  }
}
```

Anexo 101 . Inicializador y función “cargarGruposDelAlumno”.

```
Complexity is 6 It's time to do something...
cargarCuestionariosPendientes(): void {
  if (this.usuarioId !== null) {
    Complexity is 6 It's time to do something...
    this.gruposDelAlumno.forEach(grupo => {
      this.solvedQuizzesService.obtenerCuestionariosPendientes(this.usuarioId!, grupo.GrupoID).subscribe(
        Complexity is 4 Everything is cool!
        response => {
          if (response && response.data && response.data.length > 0) {
            this.cuestionariosPendientes = [...this.cuestionariosPendientes, ...response.data];
            console.log('Cuestionarios pendientes para el grupo ${grupo.GrupoID}:', response.data);
          } else {
            console.log('No se encontraron cuestionarios pendientes para el grupo ${grupo.GrupoID}.');
          }
        },
        error => {
          console.error('Error al obtener cuestionarios pendientes para el grupo ${grupo.GrupoID}:', error);
        }
      );
    });
  }
}
```

Anexo 102. Función “cargarCuestionariosPendientes”

```

    irACuestionario(cuestionarioID: number, grupoID: number) {
      console.log(`Navigating to quiz with ID: ${cuestionarioID} and group ID: ${grupoID}`);
      if (cuestionarioID && grupoID) {

        this.router.navigate(['/alumno/cuestionarios/${cuestionarioID}/preguntas'], {
          queryParams: { grupoID: grupoID }
        });
      } else {
        console.error('El ID del cuestionario o del grupo es indefinido o nulo.');
```

Anexo 103. Función “irAcuestionario”.

Para visualizar el cuestionario del usuario se crearon una serie de funciones para darle una funcionalidad correcta al cuestionario. Primero en el inicializador se obtiene el id del usuario que ha sido autenticado y el id del cuestionario mediante el id del grupo. Luego la primera función “cargarCuestionario” a partir del “cuestionarioService” de la función “getPreguntasRespuestas” se muestran las preguntas y respuestas del cuestionario mostrando 5 preguntas por página.

```

30  ngOnInit(): void {
31    this.usuarioID = this.authService.getUserId();
32    if (this.usuarioID === null) {
33      console.error('Usuario no logueado o ID de usuario no encontrado.');
```

Anexo 104. Inicializador del cuestionario y función “cargarCuestionario”.

De igual manera se creó la función para enviar las respuestas.

```
66   enviarRespuestas(event?: Event): void {
67     if (event) event.preventDefault();
68
69     const respuestas = Object.keys(this.respuestasSeleccionadas).map(preguntaId => ({
70       CuestionarioID: this.cuestionarioID,
71       PreguntaID: +preguntaId,
72       respuesta_valor: this.respuestasSeleccionadas[+preguntaId]
73     }));
74     if (this.usuarioID && this.cuestionarioID && this.grupoID) {
75       this.cuestionarioService.enviarRespuestas(this.usuarioID, respuestas).subscribe(
76         Complexity is 3 Everything is cool!
77         response => {
78           console.log('Respuestas guardadas:', response);
79           const data = {
80             UsuarioID: this.usuarioID,
81             CuestionarioID: this.cuestionarioID,
82             GrupoID: this.grupoID
83           };
84           this.solvedQuizzesService.marcarCuestionarioResuelto(data).subscribe(
85             res => {
86               console.log('Cuestionario marcado como resuelto:', res);
87               this.router.navigate(['/alumno/resultados/${this.usuarioID}/${this.cuestionarioID}']);
88             },
89             error => {
90               console.error('Error al marcar cuestionario como resuelto:', error);
91             }
92           );
93           error => {
94             console.error('Error al enviar respuestas:', error);
95           }
96         );
97       } else {
98         console.error('Faltan datos: UsuarioID, CuestionarioID o GrupoID.');
```

Anexo 105. Función “enviarRespuestas”.

```
102   paginaAnterior(): void {
103     if (this.paginaActual > 0) {
104       this.paginaActual--;
105       this.actualizarPreguntasPaginaActual();
106     }
107   }
108
109   paginaSiguiente(): void {
110     if (this.paginaActual < this.totalPaginas - 1) {
111       this.paginaActual++;
112       this.actualizarPreguntasPaginaActual();
113     }
114   }
115
116   isLastPage(): boolean {
117     return this.paginaActual === this.totalPaginas - 1;
118   }
119
120   actualizarPreguntasPaginaActual(): void {
121     this.preguntasPaginaActual = this.preguntas.slice(this.paginaActual * 5, (this.paginaActual + 1) * 5);
122   }
123 }
```

Anexo 106. Funciones para la navegación del cuestionario.

```

<div class="container my-5">
  <div class="text-center mb-4">
    <div class="title-bar d-flex align-items-center justify-content-between">
      <h1 class="display-4">{{ cuestionario.nombre }}</h1>
      
    </div>
    <br>
    <h2 class="lead">{{ cuestionario.descripcion }}</h2>
  </div>
  <form (ngSubmit)="enviarRespuestas($event)">
    <div *ngFor="let pregunta of preguntasPaginaActual" class="mb-4">
      <div class="card p-3 mb-3">
        <h3 class="card-title">{{ pregunta.Texto }}</h3>
        <div class="form-group">
          <div *ngFor="let respuesta of pregunta.respuestas">
            <div class="form-check">

```

Anexo 107. Código de la vista del cuestionario parte 1.

```

      <input
        type="radio"
        class="form-check-input"
        [name]="pregunta + pregunta.PreguntaID"
        [value]="respuesta.Valor"
        (change)="onAnswerChange(pregunta.PreguntaID, respuesta.Valor)"/>
      <label class="form-check-label">{{ respuesta.Texto }}</label>
    </div>
  </div>
</div>
</div>
</div>
</div>
</div>
<div class="d-flex justify-content-between">
  <button type="button" class="btn btn-secondary" (click)="paginaAnterior()" [disabled]="paginaActual === 0">Anterior</button>
  <button *ngIf="isLastPage(); else siguienteButton" type="button" (click)="enviarRespuestas()" class="btn btn-success">Enviar</button>
  <ng-template #siguienteButton>
    <button type="button" class="btn btn-primary" (click)="paginaSiguiente()" [disabled]="paginaActual >= totalPaginas - 1">Siguiente</button>
  </ng-template>
</div>
</form>
</div>

```

Anexo 108. Código de la vista del cuestionario parte 2.

Para visualizar los resultados del alumno se creó la vista de los resultados mostrando la información del alumno, tales como: Nombre, apellidos, correo, semestre y carrera. De igual manera contiene los puntajes obtenidos para cada variable y una gráfica para una mejor retroalimentación.

```

Go to component | You, hace 4 segundos | 2 authors (Fernanda-Chable and one other)
1 <div class="container my-5">
2   <div class="datos-student">
3     <div *ngIf="alumno">
4       <h2>Resultados de {{ alumno.Nombre }} {{ alumno.Primer_apellido }} {{ alumno.Segundo_apellido }} ({{ alumno.Matricula }})</h2>
5       <p><strong>Nombre:</strong> {{ alumno.Nombre }} {{ alumno.Primer_apellido }} {{ alumno.Segundo_apellido }}</p>
6       <p><strong>Correo:</strong> {{ alumno.Correo }}</p>
7       <p><strong>Semestre:</strong> {{ alumno.Semestre }}</p>
8       <p><strong>Carrera:</strong> {{ alumno.CarreraID.Nombre }}</p>
9     </div>
10  </div>

```

Anexo 109. Código de la vista de resultados del cuestionario parte 1

```

<div class="text-center mb-4">
  <h2 class="display-5">Resultado de autoevaluación</h2>
</div>
<div class="row">
  <div class="col-md-8 offset-md-2">
    <div *ngIf="hasPuntajes">
      <div class="table-container">
        <table class="table table-striped table-sm">
          <thead>
            <tr>
              <th>Variable</th>
              <th>Puntaje</th>
            </tr>
          </thead>
          <tbody>
            <tr>
              <td>Factor de corrección</td>
              <td>{{ puntajes.puntaje_variable_0 }}</td>
            </tr>
            <tr>
              <td>Buscar y aprovechar las oportunidades</td>
              <td>{{ puntajes.puntaje_variable_1 }}</td>
            </tr>
            <tr>
              <td>Ser Persistentes</td>
              <td>{{ puntajes.puntaje_variable_2 }}</td>
            </tr>
            <tr>
              <td>Cumplimiento de compromisos</td>
              <td>{{ puntajes.puntaje_variable_3 }}</td>
            </tr>
            <tr>
              <td>Exigir eficiencia y calidad</td>
              <td>{{ puntajes.puntaje_variable_4 }}</td>
            </tr>
            <tr>
              <td>Correr riesgos moderados</td>
              <td>{{ puntajes.puntaje_variable_5 }}</td>
            </tr>
          </tbody>
        </table>
      </div>
    </div>
  </div>
</div>

```

Anexo 110. Código de la vista de resultados del cuestionario parte 2

```

      <tr>
        <td>Fijar Objetivos</td>
        <td>{{ puntajes.puntaje_variable_6 }}</td>
      </tr>
      <tr>
        <td>Buscar y conseguir información</td>
        <td>{{ puntajes.puntaje_variable_7 }}</td>
      </tr>
      <tr>
        <td>Planificar y hacer seguimiento sistemático</td>
        <td>{{ puntajes.puntaje_variable_8 }}</td>
      </tr>
      <tr>
        <td>Persuasión y creación de redes de apoyo</td>
        <td>{{ puntajes.puntaje_variable_9 }}</td>
      </tr>
      <tr>
        <td>Tener autoconfianza</td>
        <td>{{ puntajes.puntaje_variable_10 }}</td>
      </tr>
    </tbody>
  </table>
</div>
<!-- Gráfico de radar -->
<h2 class="display-5 text-center">Gráfico de autoevaluación</h2>
<div class="text-center">
  <canvas id="graficaAraña" width="800" height="0"></canvas>
</div>
</div>
<div *ngIf="!hasPuntajes">
  <p class="text-center">No se encontraron datos para el cuestionario seleccionado.</p>
</div>
</div>
</div>

```

Anexo 111. Código de la vista de resultados del cuestionario parte 3

Para darle funcionalidad a la vista de resultados del alumno se inicializan el id de usuario y el id de cuestionario, el cual llama el servicio “getStudent” para obtener la información del alumno mediante su id, para posteriormente almacenar la información en una variable data.

```
ngOnInit(): void {
  const usuarioID = this.route.snapshot.paramMap.get('usuarioID');
  const cuestionarioID = this.route.snapshot.paramMap.get('cuestionarioID');
  console.log('usuarioID:', usuarioID);
  console.log('cuestionarioID:', cuestionarioID);
  if (!usuarioID || !cuestionarioID) {
    console.error('ID de usuario o CuestionarioID no válidos:', { usuarioID, cuestionarioID });
    return;
  }
  const usuarioIDNumerico = +usuarioID;
  const cuestionarioIDNumerico = +cuestionarioID;
  // Llamar al servicio para obtener la información del alumno por ID
  this.userService.getStudent(usuarioIDNumerico).subscribe(
    (data) => {
      this.alumno = data.data; // Guardar la información del alumno en la variable
      console.log('Información del alumno:', this.alumno);
    },
    (error) => {
      console.error('Error al obtener los datos del alumno:', error);
    }
  );
}
```

Anexo 112. Inicializador de id de usuario y cuestionario.

De igual manera se obtiene los puntajes a partir del “servicioService” con la función “obtenerPuntajesPorCuestionario” mediante el id de usuario y el cuestionario, también se le agrego un mensaje de error en caso de no obtener puntajes validos o al no recibir los puntajes por problemas del servicio.

```
this.puntajeService.obtenerPuntajesPorCuestionario(usuarioIDNumerico, cuestionarioIDNumerico).subscribe(
  Complexity is 15 You must be kidding
  data => {
    console.log('Datos obtenidos:', JSON.stringify(data, null, 2));
    if (data && Object.keys(data).length > 0) {
      this.puntajes = {
        puntaje_variable_0: Number(data.puntaje_variable_0) || 0,
        puntaje_variable_1: Number(data.puntajes_variables.puntaje_variable_1) || 0,
        puntaje_variable_2: Number(data.puntajes_variables.puntaje_variable_2) || 0,
        puntaje_variable_3: Number(data.puntajes_variables.puntaje_variable_3) || 0,
        puntaje_variable_4: Number(data.puntajes_variables.puntaje_variable_4) || 0,
        puntaje_variable_5: Number(data.puntajes_variables.puntaje_variable_5) || 0,
        puntaje_variable_6: Number(data.puntajes_variables.puntaje_variable_6) || 0,
        puntaje_variable_7: Number(data.puntajes_variables.puntaje_variable_7) || 0,
        puntaje_variable_8: Number(data.puntajes_variables.puntaje_variable_8) || 0,
        puntaje_variable_9: Number(data.puntajes_variables.puntaje_variable_9) || 0,
        puntaje_variable_10: Number(data.puntajes_variables.puntaje_variable_10) || 0,
      };
      this.hasPuntajes = true;
      setTimeout(() => this.createRadarChart(), 100);
    } else {
      console.warn('No se recibieron puntajes válidos', data);
      this.hasPuntajes = false;
    }
  },
  error => {
    console.error('Error al obtener puntajes:', error);
  }
);
}
```

Anexo 113. Obtener puntajes del cuestionario

Para crear la gráfica de telaraña se realizó la función “createRadarChart” el cual verifica que los puntajes obtenidos sean válidos para cada variable asignando el tipo de gráfico que se va a dibujar, configurando de igual manera el rango máximo y mínimo para mostrar los datos. Si en caso de que los datos no son válidos, la función muestra un mensaje de error en la consola.

```
createRadarChart(): void {
  if (this.puntajes && this.hasPuntajes) {
    const data = [
      this.puntajes.puntaje_variable_0,
      this.puntajes.puntaje_variable_1,
      this.puntajes.puntaje_variable_2,
      this.puntajes.puntaje_variable_3,
      this.puntajes.puntaje_variable_4,
      this.puntajes.puntaje_variable_5,
      this.puntajes.puntaje_variable_6,
      this.puntajes.puntaje_variable_7,
      this.puntajes.puntaje_variable_8,
      this.puntajes.puntaje_variable_9,
      this.puntajes.puntaje_variable_10
    ];
    const ctx = document.getElementById('graficaAraña') as HTMLCanvasElement;
    if (ctx) {
      if (this.chart) {
        this.chart.destroy();
      }
      this.chart = new Chart(ctx, {
        type: 'radar',
        data: {
          labels: [
            'Factor de corrección',
            'Buscar y aprovechar las oportunidades',
            'Ser Persistentes',
            'Cumplimiento de compromisos',
            'Exigir eficiencia y calidad',
            'Correr riesgos moderados',
            'Fijar Objetivos',
            'Buscar y conseguir información',
            'Planificar y hacer seguimiento sistemático',
            'Persuasión y creación de redes de apoyo',
            'Tener autoconfianza'
          ],
          datasets: data
        }
      });
    }
  }
}
```

Anexo 114. Función “createRadarChar”t parte 1

```

120     datasets: [{
121         label: 'Puntajes por variable',
122         data: data,
123         fill: true,
124         backgroundColor: 'rgba(34, 202, 236, 0.2)',
125         borderColor: 'rgb(34, 202, 236)',
126         pointBackgroundColor: 'rgb(34, 202, 236)',
127         pointBorderColor: '#fff',
128         pointHoverBackgroundColor: '#fff',
129         pointHoverBorderColor: 'rgb(34, 202, 236)'
130     }]
131 },
132 options: {
133     scales: {
134         r: {
135             suggestedMin: 0,
136             suggestedMax: 30,
137             ticks: {
138                 stepSize: 5
139             }
140         }
141     }
142 });
143 } else {
144     console.error('No se encontró el contexto del canvas para el gráfico.');
```

Anexo 115. Función “createRadarChar”t parte 2.

Para el maestro se creó su propio menú este de igual manera que el Dashboard del alumno obtiene la información del usuario, pero en este caso cambia las secciones del menú. Ahora las secciones que contiene es el Home, grupos, cuestionarios aplicados y alumnos.

```
<div *ngIf="isLoggedIn" class="dashboard">
  <div class="sidebar">
    <div class="user-info">
      
      <p *ngIf="teacher">{{ teacher.Nombre }} {{ teacher.Primer_apellido }} {{ teacher.Segundo_apellido }}</p>
      <p *ngIf="teacher">{{ teacher.Correo }}</p>
    </div>
    <div class="menu">
      <a [routerLink]="['/maestro/dashTeacher']">
        <mat-icon>home</mat-icon> Home
      </a>
      <a [routerLink]="['/maestro/gruposTeacher']">
        <mat-icon>groups</mat-icon> Grupos
      </a>
      <a [routerLink]="['/maestro/applied-quiz']">
        <mat-icon>fact_check</mat-icon> Cuestionarios aplicados
      </a>
      <a [routerLink]="['/maestro/listAlumnos']">
        <mat-icon>contacts_product</mat-icon> Alumnos
      </a>
      <a (click)="logout()" *ngIf="isLoggedIn">
        <mat-icon>logout</mat-icon> Cerrar Sesión
      </a>
    </div>
  </div>
  <div class="main-content">
    <div class="content-container">
      <router-outlet></router-outlet>
    </div>
  </div>
</div>
```

Anexo 116. Código de la vista del menú del maestro.

Como en el caso del alumno se usa la misma estructura para obtener la información del maestro, mediante el id de usuario y también incluye la función para cerrar la sesión.

```
ngOnInit(): void {
  this.isLoggedIn = this.authService.isLoggedIn();

  if (this.isLoggedIn) {
    this.loadTeacherData();
  } else {
    console.error('El usuario no está autenticado');
  }
}

Complexity is 5 Everything is cool!
loadTeacherData(): void {
  const teacherId = this.authService.getUserId();
  console.log('ID del maestro:', teacherId);
  if (teacherId) {
    this.userService.getTeacher(teacherId).subscribe(
      response => {
        console.log('Respuesta del API:', response);
        if (response.data) {
          this.teacher = response.data;
          console.log('Datos del maestro:', this.teacher);
        } else {
          console.error('No se encontraron datos del maestro');
        }
      },
      error => {
        console.error('Error al cargar datos del maestro', error);
      }
    );
  } else {
    console.error('No se encontró un ID de usuario autenticado');
  }
}

// Método para cerrar sesión
logout(): void {
  this.authService.logout();
  this.isLoggedIn = false;
  this.router.navigate(['/login']);
}
```

Anexo 117. Funciones para el menú del maestro.

El maestro podrá crear grupo con la siguiente función la cual al llenar el formulario con la función “onSubmit” el realiza una subscripción al servicio “createGroup” del “teacherGroupsService” generando un código de acceso aleatoriamente, si el proceso se ha realizado correctamente muestre un mensaje indicando que el grupo de ha creado exitosamente y en caso de que no, muestre el error.

```
Complexity is 7 It's time to do something...
onSubmit() {
  if (this.groupForm.valid) {
    const groupData = {
      ...this.groupForm.value,
      usuario_id: this.userid
    };
    this.teacherGroupsService.createGroup(groupData).subscribe({
      Complexity is 4 Everything is cool!
      next: (response) => {
        console.log('Grupo creado exitosamente:', response);
        const codigoAcceso = response.data?.codigo_acceso;
        console.log('Codigo de acceso', codigoAcceso);
        if (codigoAcceso) {
          Swal.fire({
            title: '¡Grupo Creado Exitosamente!',
            text: `El código de acceso es: ${codigoAcceso}`,
            icon: 'success',
            confirmButtonText: 'Aceptar'
          }).then((result) => {
            if (result.isConfirmed) {
              this.router.navigate(['/maestro/createGroup']);
            }
          });
        } else {
          this.snackBar.open('Grupo creado, pero no se obtuvo el código de acceso.', 'Cerrar', {
            duration: 5000,
          });
        }
        this.groupForm.reset();
      },
    });
  }
}
```

Anexo 118. Función almacenar datos del grupo parte 1.

```
error: (error) => {
  console.error('Error al crear el grupo', error);
  this.snackBar.open('Error al crear el grupo. Inténtalo de nuevo.', 'Cerrar', {
    duration: 5000,
  });
}
});
} else {
  this.snackBar.open('Por favor, completa todos los campos requeridos.', 'Cerrar', {
    duration: 5000,
  });
}
}
```

Anexo 119. Función almacenar datos del grupo parte 1.

El maestro tiene la función de publicar cuestionarios a sus grupos por ello se creó el componente para dicha función. Para el funcionamiento del componente se inicializa el id del maestro para obtener sus grupos, de igual manera los cuestionarios existentes, posteriormente se cargan las preguntas del cuestionario.

```
34   ngOnInit() {
35     const usuarioID = this.authService.getUserId();
36     if (usuarioID !== null) {
37       this.obtenerCuestionarios();
38       this.obtenerGrupos(usuarioID);
39     } else {
40       console.error('No se encontró el ID de usuario logueado');
41     }
42   }
43   ,
44   obtenerCuestionarios() {
45     this.cuestionarioService.getQuizVersions().subscribe((data: any) => {
46       console.log('Cuestionarios:', data.cuestionarios);
47       this.cuestionarios = Array.isArray(data.cuestionarios) ? data.cuestionarios : [];
48     });
49   }
50   ,
51   obtenerGrupos(usuarioID: number) {
52     this.teacherGroupsService.getGruposDeMaestro(usuarioID).subscribe((data: any) => {
53       console.log('Grupos:', data);
54       this.grupos = Array.isArray(data) ? data : [];
55     });
56   }
57   ,
58   cargarPreguntas() {
59     if (this.cuestionarioSeleccionado) {
60       this.cuestionarioService.getPreguntasRespuestas(this.cuestionarioSeleccionado).subscribe((data: any) => {
61         console.log('Preguntas:', data.preguntas);
62         this.preguntas = data.preguntas;
63       });
64     }
65   }
66   ,
67 }
```

Anexo 120. Inicializador, funciones “obtenerCuestionarios”, “obtenerGrupos” y “cargarPreguntas”.

Para publicar el cuestionario se realizó la función “publicarCuestionario” en ella se realiza la condición si la fecha de publicación y fecha de cierre han sido establecidas, obtiene el id del grupo al cual se va a publicar y las fechas anteriormente para realizar la subscripción al “cuestionarioService” para la función “publishQuiz”, si el cuestionario se publica correctamente se mostrará un mensaje el cual indica que el cuestionario se publicó correctamente y si no muestra el mensaje error al publicar el cuestionario. También se puede cancelar el proceso.

```

publicarCuestionario() {
  if (!this.fechaPublicacion || !this.fechaCierre) {
    alert('Por favor, completa ambas fechas antes de publicar.');
```

`return;`
 }
 const payload = {
 grupo_id: this.grupoSeleccionado,
 fecha_publicacion: this.fechaPublicacion,
 fecha_cierre: this.fechaCierre,
 };

 this.cuestionarioService.publishQuiz(this.cuestionarioSeleccionado, payload).subscribe({
 next: () => {
 alert('Cuestionario publicado exitosamente');
 this.router.navigate(['/maestro/quiz']);
 },
 error: (err) => {
 console.error('Error al publicar el cuestionario:', err);
 alert('Hubo un error al publicar el cuestionario.');
`},`
 });
}

cancelar() {
 this.router.navigate(['/maestro/quiz']);
}

Anexo 121. Función “publicarCuestionario”.

El maestro Puede visualizar la información del grupo con sus integrantes del grupo, de igual manera los cuestionarios que han sido aplicados en el siguiente componente.

```

1  Go to component | You have 48 segundos | 2 authors (Fernanda-Chable and one other)
2  <div *ngIf="grupo">
3    <section class="quiz-details">
4      <div class="quiz-header">
5        <h2><strong>Carrera:</strong> {{ grupo.carrera.Nombre }}</h2>
6        <h2><strong>Grupo:</strong> {{ grupo.Nombre }}&nbsp;<strong>Semestre:</strong> {{ grupo.semestre }}</h2>
7        <h2><strong>Código de acceso:</strong> {{ grupo.codigo_acceso }}</h2>
8      </div>
9
10     <div class="group-members">
11       <h3>Integrantes del Grupo</h3>
12       <table>
13         <thead>
14           <tr>
15             <th>Nombre</th>
16             <th>Primer Apellido</th>
17             <th>Segundo Apellido</th>
18             <th>Carrera</th>
19           </tr>
20         </thead>
21         <tbody>
22           <tr *ngFor="let integrante of integrantes">
23             <td>{{ integrante.Nombre }}</td>
24             <td>{{ integrante.Primer_apellido }}</td>
25             <td>{{ integrante.Segundo_apellido }}</td>
26             <td>{{ integrante.Carrera.Nombre }}</td>
27           </tr>
28         </tbody>
29       </table>
30     </div>
31
32     <div class="quiz-list">
33       <h3>Cuestionarios Aplicados</h3>
34       <ul>
35         <li *ngFor="let cuestionario of cuestionarios" (click)="selectCuestionario(cuestionario)">
36           <p><strong>Nombre:</strong> {{ cuestionario.Nombre }}</p>
37           <p><strong>Detalles de Cuestionario:</strong> {{ cuestionario.Descripcion }}</p>
38         </li>
39       </ul>
40     </div>
41

```

Anexo 122. Código la vista de información del cuestionario aplicado

Se crearon varias funciones para la vista de información del cuestionario aplicado al grupo, al acceder a la vista se obtiene el id del grupo, de igual manera se inicializan las funciones para obtener detalles del grupo, obtener cuestionarios aplicados y obtener miembros del grupo.

```
36   ngOnInit(): void {
37     this.grupoId = +this.route.snapshot.paramMap.get('grupoId')!;
38     this.getGroupDetails();
39     this.getAppliedQuizzes();
40     this.getGroupMembers();
41   }
42
43   Complexity is 3 Everything is cool!
44   getGroupDetails() {
45     this.teacherGroupsService.getGroupById(this.grupoId).subscribe(
46       (response) => {
47         this.grupo = response.data;
48       },
49       (error) => {
50         console.error('Error al obtener los detalles del grupo:', error);
51       }
52     );
53   }
54   Complexity is 3 Everything is cool!
55   getAppliedQuizzes() {
56     this.teacherGroupsService.getAppliedQuizzesByGroup(this.grupoId).subscribe(
57       (response) => {
58         this.cuestionarios = response.data;
59       },
60       (error) => {
61         console.error('Error al obtener los cuestionarios aplicados:', error);
62       }
63     );
64   }
65   Complexity is 3 Everything is cool!
66   getGroupMembers() {
67     this.teacherGroupsService.getIntegrantes(this.grupoId).subscribe(
68       (response) => {
69         this.integrantes = response.data;
70       },
71       (error) => {
72         console.error('Error al obtener los integrantes del grupo:', error);
73       }
74     );
75   }
```

Anexo 123. Inicializador y funciones “getGroupDetails”, “getAppliedQuizzes” y “getGroupMembers”.

Para el Dashboard de administrador Funciona de la misma manera que el de maestro y alumno obteniendo la información del administrador, pero en este caso las secciones que posee son: Países, Instituciones, Administración de cuestionarios, Administración de usuarios, Cuestionarios aplicados y Edición de la página principal.

```
Go to component | You, el mes pasado | 1 author (You)
1 <div class="main-content">
2   <header class="header">
3     <div class="header-box">
4       <h1>Bienvenido, {{ currentDate }}</h1>
5     </div>
6     <div class="search-and-button">
7       <input type="search" placeholder="Search">
8     </div>
9   </header>
10
11   <section class="home">
12     <div class="home-box">
13       <h1><mat-icon>home</mat-icon> HOME</h1>
14     </div>
15   </section>
16
17   <div class="buttons">
18     <button class="button blue" [routerLink]="['/administrador/lista-estudiantes']">
19       <mat-icon>account_circle</mat-icon> Alumnos <span class="counter">{{ totalAlumnos }}</span>
20     </button>
21     <button class="button orange" [routerLink]="['/administrador/lista-docentes']">
22       <mat-icon>account_circle</mat-icon> Maestros <span class="counter">{{ totalMaestros }}</span>
23     </button>
24     <button class="button orange" [routerLink]="['/administrador/lista-docentes']">
25       <mat-icon>account_circle</mat-icon> Informes <span class="counter">{{ totalMaestros }}</span>
26     </button>
27     <button class="button green" [routerLink]="['/administrador/instituciones']">
28       <mat-icon>check_circle</mat-icon> Instituciones <span class="counter">{{ totalInstituciones }}</span>
29     </button>
30     <button class="button pink" [routerLink]="['/alumno/pendiente']">
31       <mat-icon>text_snippet</mat-icon> Cuestionarios aplicados <span class="counter">{{ totalCuestionariosPendientes }}</span>
32     </button>
33   </div>
34 </div>
```

Anexo 124. Código de la vista del Dashboard del administrador.

```
Go to component | You, anteyar | 1 author (You)
1 <div *ngIf="isLoggedIn" class="dashboard">
2   <div class="sidebar">
3     <div class="user-info" (click)="navigateToProfile()">
4       
5       <p *ngIf="Administrator">{{ Administrator.Nombre }} {{ Administrator.Primer_apellido }} {{ Administrator.Segundo_apellido }}</p>
6       <p *ngIf="Administrator">{{ Administrator.Correo }}</p>
7     </div>
8     <div class="menu">
9       <a [routerLink]="['/administrador/dash-admin']"> You, el mes pasado • Frontend Creación del Dashboard de administrador...
10       <mat-icon>home</mat-icon> Home
11     </a>
12     <a [routerLink]="['/administrador/paises']">
13       <mat-icon>language</mat-icon> Países
14     </a>
15     <a [routerLink]="['/administrador/instituciones']">
16       <mat-icon>groups</mat-icon> Instituciones
17     </a>
18     <a [routerLink]="['/administrador/quiz']">
19       <mat-icon>quiz</mat-icon> Administracion de cuestionarios
20     </a>
21     <a [routerLink]="['/administrador/usuarios']">
22       <mat-icon>groups</mat-icon> Administracion de usuarios
23     </a>
24     <a [routerLink]="['']">
25       <mat-icon>view_timeline</mat-icon> Cuestionarios aplicados
26     </a>
27     <a [routerLink]="['']">
28       <mat-icon>edit_square</mat-icon> Edición de la página principal
29     </a>
30     <a (click)="logout()" *ngIf="isLoggedIn">
31       <mat-icon>logout</mat-icon> Cerrar Sesión
32     </a>
33   </div>
34 </div>
```

Anexo 125. Código de la vista del menú del administrador.

En el apartado de países se creó una función a partir del servicio “getCountries” de catalogs el cual se cargan todos los países registrados en el sistema, también se puede obtener un país con la función “filterCountries”.

```

Complexity is 4 Everything is cool!
30 loadCountries(): void {
31   this.catalogs.getCountries().subscribe({
32     next: (response) => {
33       this.countries = response.data || [];
34       this.filteredCountries = this.countries;
35     },
36     error: (error) => {
37       console.error('Error al cargar los países', error);
38     }
39   });
40 }
41
42 goToInstitutions(countryId: number): void {
43   this.router.navigate(['/administrador/institucionesf', countryId]);
44 }
Complexity is 3 Everything is cool!
45 filterCountries(): void {
46   // Filtrar la lista de países en base a la consulta de búsqueda
47   this.filteredCountries = this.countries.filter(country =>
48     this.searchQuery ? country.Nombre.toLowerCase().includes(this.searchQuery.toLowerCase()) : true
49   );
50 }
51

```

Anexo 126. Funciones “loadCountries” y “filteredCountries”.

Para agregar un país se creó un formulario el cual contiene una suscripción al servicio “CreateCountries” agregando el id del país y su nombre. Si el país se agregó correctamente mostrará el mensaje del país agregado y si no mostrará el error obtenido. Se agregó una función para cancelar el proceso.

```

20 constructor(
21   private fb: FormBuilder,
22   private catalogs: CatalogsService,
23   public dialogRef: MatDialogRef<AddCountryComponent>
24 ) {
25   this.countryForm = this.fb.group({
26     Nombre: ['', Validators.required]
27   });
28 }
29
Complexity is 5 Everything is cool!
30 onSubmit(): void {
31   if (this.countryForm.valid) {
32     const newCountry: CreateCountries = { IDPaís: 0, Nombre: this.countryForm.value.Nombre };
33     this.catalogs.createCountry(newCountry).subscribe({
34       next: (response) => {
35         console.log('País agregado:', response);
36         this.dialogRef.close(true);
37       },
38       error: (error: any) => {
39         console.error('Error al agregar país:', error);
40       }
41     });
42   }
43 }
44
45 onCancel(): void {
46   this.dialogRef.close(false);
47 }
48

```

Anexo 127. Código componente agregar país

De la misma forma para la lista de países se realizó el proceso para las instituciones. En este caso se añadió un filtro para obtener las instituciones filtradas por país.

```
26   ngOnInit(): void {
27     this.loadInstitutions();
28     this.loadCountries();
29   }
30   loadInstitutions(): void {
31     this.catalogs.getInstitutions().subscribe({
32       next: (response) => {
33         this.institutions = response.data || [];
34         this.filteredInstitutions = this.institutions;
35       },
36       error: (error) => {
37         console.error('Error al cargar las instituciones', error);
38       }
39     });
40   }
41   loadCountries(): void {
42     this.catalogs.getCountries().subscribe({
43       next: (response) => {
44         this.countries = response.data || [];
45       },
46       error: (error) => {
47         console.error('Error al cargar los países', error);
48       }
49     });
50   }
```

Anexo 128. Funcionalidad de la vista lista de instituciones.

```
51   filterInstitutions(): void {
52     if (this.selectedCountryId) {
53       // Si hay un país seleccionado, utilizar el servicio para obtener solo las instituciones de ese país
54       this.catalogs.getInstitutionsId(this.selectedCountryId).subscribe({
55         next: (response) => {
56           // Filtrar por nombre en base al query de búsqueda localmente
57           this.filteredInstitutions = response.data?.filter(institution =>
58             this.searchQuery ? institution.Nombre.toLowerCase().includes(this.searchQuery.toLowerCase()) : true
59           ) || [];
60         },
61         error: (error) => {
62           console.error('Error al cargar las instituciones filtradas', error);
63           this.filteredInstitutions = [];
64         }
65       });
66     } else {
67       // Si no hay un país seleccionado, utilizar el servicio getInstitutions() para obtener todas las instituciones
68       this.catalogs.getInstitutions().subscribe({
69         next: (response) => {
70           // Filtrar por nombre en base al query de búsqueda localmente
71           this.filteredInstitutions = response.data?.filter(institution =>
72             this.searchQuery ? institution.Nombre.toLowerCase().includes(this.searchQuery.toLowerCase()) : true
73           ) || [];
74         },
75         error: (error) => {
76           console.error('Error al cargar todas las instituciones', error);
77           this.filteredInstitutions = [];
78         }
79       });
80     }
81   }
```

Anexo 129. Función de Instituciones filtradas por país.

Para obtener los maestros de una institución, se crearon unas funciones para obtener los maestros de la institución.

```
export class ListTeachersInstitutionsComponent {
  teachers: getTeachers[] = [];
  InstitutionID: string | null = null;
  institution: Institutions | undefined;

  constructor(private router: Router, private route: ActivatedRoute,
    private userService: UserService, private catalogs: CatalogsService,
    private location: Location){}

  ngOnInit(): void {
    this.InstitutionID = this.route.snapshot.paramMap.get('id');
    if(this.InstitutionID){
      this.loadTeachersInst(this.InstitutionID);
      this.loadInstitution(this.InstitutionID);
    }
  }

  loadTeachersInst(institutionID: string): void {
    this.userService.getTeachersInst(institutionID).subscribe(response => {
      this.teachers = response.data;
    });
  }

  loadInstitution(InstitutionID: string): void {
    this.catalogs.getInstitution(InstitutionID).subscribe({
      next: (response: any) => {
        console.log('API Response:', response);
        this.institution = response.data.institution;
      },
      error: (error) => {
        console.error('Error al cargar la institución', error)
      }
    });
  }

  goBack(): void {
    this.location.back();
  }
}
```

Anexo 130. Funcionalidad de la lista de maestros por institución.

Para agregar un maestro a una institución se usa la misma estructura que el registro de alumnos, pero en este caso el formulario se tiene por preestablecido el tipo de usuario como Maestro y de igual manera el semestre se mantiene en nulo ya que un maestro no posee un semestre en específico.

```
this.formGroup = new FormGroup({
  nombre: new FormControl('', [nameValidator, makeRequired, lengthValidator({ min: 1, max: 30 })]),
  papellido: new FormControl('', [nameValidator, makeRequired, lengthValidator({ min: 1, max: 15 })]),
  sapellido: new FormControl('', [nameValidator, makeRequired, lengthValidator({ min: 1, max: 15 })]),
  matricula: new FormControl('', [makeRequired, lengthValidator({ min: 1, max: 15 })]),
  semestre: new FormControl('Null', []),
  tipo: new FormControl('Maestro', []),
  correo: new FormControl('', [emailValidator(), makeRequired, lengthValidator({ min: 1, max: 30 })]),
  contraseña: new FormControl('', [makeRequired, lengthValidator({ min: 8, max: 15 })]),
  confirmcontraseña: new FormControl('', [makeRequired, lengthValidator({ min: 8, max: 15 })]),
  edad: new FormControl('', [makeRequired]),
  genero: new FormControl('', [makeRequired]),
  institucion: new FormControl('', [makeRequired]),
  pais: new FormControl('', [makeRequired]),
  carrera: new FormControl('', []),
}, { validators: confirmPassword });

this.loadCountriesAndProcessParams();
this.formGroup.get('pais')?.valueChanges.subscribe(selectedCountryId => {
  this.onCountryChange(selectedCountryId);
});
this.ages = Array.from({ length: 99 - 17 + 1 }, (_, i) => i + 17);
}
```

Anexo 131. Código del formulario del maestro.

Para el caso de obtener la lista de integrantes del grupo del maestro, se inicializa el id del maestro y el id del grupo para obtener los miembros del grupo. Se crearon 2 funciones las cuales son para obtener los datos del maestro y cargar los integrantes del grupo a partir de los servicios “getTeachers” y “getIntegrantes”.

```

Complexity is 3 Everything is cool!
ngOnInit(): void {
  this.grupoId = this.route.snapshot.paramMap.get('grupoId');
  const usuarioId = this.route.snapshot.queryParamMap.get('usuarioId');

  if (this.grupoId) {
    this.loadGroupMembers(Number(this.grupoId));
    this.loadGroupDetails(Number(this.grupoId));
  }
  if (usuarioId) {
    this.loadTeacherDetails(Number(usuarioId));
  }
}

Complexity is 4 Everything is cool!
loadTeacherDetails(usuarioId: number): void {
  this.userService.getTeacher(usuarioId).subscribe({
    next: (response: any) => {
      this.teacher = response.data;
    },
    error: (error) => {
      console.error('Error al cargar los detalles del maestro', error);
    }
  });
}

Complexity is 5 Everything is cool!
loadGroupMembers(grupoId: number): void {
  Complexity is 4 Everything is cool!
  this.teacherGroupsService.getIntegrantes(grupoId).subscribe(response => {
    console.log('Respuesta completa del servicio:', response);
    this.students = response.data.map((student: any) => ({
      ...student,
      semestre: student.Semestre,
    }));

    const institutionID = response.data[0]?.Carrera?.Institucion?.InstitucionID;
    if (institutionID) {
      this.loadInstitution(institutionID.toString());
    } else {
      console.error('No se pudo encontrar el InstitucionID en los datos del grupo');
    }
  });
}

```

Anexo 132. Funcionalidad para obtener los datos del maestro y los miembros del grupo.

Para la lista de alumnos por institución se cargan los estudiantes que pertenecen a la institución a partir del servicio “getStudentsInst” de userService.

```
loadStudentInst(institutionID: string): void {
  this.userService.getStudentsInst(institutionID).subscribe(response => {
    this.students = response.data;
    this.filteredStudents = this.students; // Inicializa con todos los estudiantes
  });
}
```

Anexo 133. Funcionalidad de lista de alumnos de la institución.

Para obtener todos los alumnos del sistema, se inicializó el servicio “getStudents” del “userService”. En este caso se agregó una función para filtrar los estudiantes por carrera, institución o país. También se agregó una función para eliminar un estudiante del sistema.

```
ngOnInit(): void {
  this.userService.getStudents().subscribe(response => {
    this.students = response.data;
    this.filteredStudents = [...this.students];
    this.extractUniqueFilters();
  });
}
Complexity is 4 Everything is cool!
extractUniqueFilters(): void {
  this.uniqueCarreras = [...new Set(this.students.map(student => student.CarreraID.Nombre))];
  this.uniqueInstituciones = [...new Set(this.students.map(student => student.InstitucionID.Nombre))];
  this.uniquePaises = [...new Set(this.students.map(student => student.IDPais.Nombre))];
}
Complexity is 7 It's time to do something...
filterStudents(): void {
  Complexity is 6 It's time to do something...
  this.filteredStudents = this.students.filter(student =>
    (this.selectedCarrera ? student.CarreraID.Nombre === this.selectedCarrera : true) &&
    (this.selectedInstitucion ? student.InstitucionID.Nombre === this.selectedInstitucion : true) &&
    (this.selectedPais ? student.IDPais.Nombre === this.selectedPais : true)
  );
}
Complexity is 6 It's time to do something...
removeStudent(UsuarioID: number): void {
  if (confirm('¿Estás seguro de que deseas eliminar este estudiante?')) {
    this.userService.deleteStudent(UsuarioID).subscribe({
      Complexity is 3 Everything is cool!
      next: () => {
        // Actualizar la lista de estudiantes después de eliminar
        this.students = this.students.filter(student => student.UsuarioID !== UsuarioID);
        this.filteredStudents = this.filteredStudents.filter(student => student.UsuarioID !== UsuarioID);
      },
      error: (error) => {
        console.error('Error al eliminar el estudiante:', error);
        alert('Hubo un error al intentar eliminar el estudiante. Por favor, intenta de nuevo.');
```

Para la administración de maestros del sistema, se realizó de la misma forma que los alumnos por lo que se inicializó el servicio “getTeachers” del “userService”. Se agregó una función para filtrar los maestros por institución o país. También se agregó una función para eliminar un maestro del sistema.

```
ngOnInit(): void {
  this.userService.getTeachers().subscribe(response => {
    this.teachers = response.data;
    this.filteredTeachers = [...this.teachers];
    this.extractUniqueFilters();
  });
}
Complexity is 3 Everything is cool!
extractUniqueFilters(): void {
  this.uniqueInstituciones = [...new Set(this.teachers.map(teacher => teacher.InstitucionID.Nombre))];
  this.uniquePaises = [...new Set(this.teachers.map(teacher => teacher.IDPais.Nombre))];
}
Complexity is 5 Everything is cool!
filterTeachers(): void {
  Complexity is 4 Everything is cool!
  this.filteredTeachers = this.teachers.filter(teacher =>
    (this.selectedInstitucion ? teacher.InstitucionID.Nombre === this.selectedInstitucion : true) &&
    (this.selectedPais ? teacher.IDPais.Nombre === this.selectedPais : true)
  );
}
Complexity is 7 It's time to do something...
removeTeacher(UsuarioID: number): void {
  if (confirm('¿Estás seguro de que deseas eliminar este docente?')) {
    this.userService.deleteTeacher(UsuarioID).subscribe({
      Complexity is 3 Everything is cool!
      next: () => {
        this.teachers = this.teachers.filter(teacher => teacher.UsuarioID !== UsuarioID);
        this.filteredTeachers = this.filteredTeachers.filter(teacher => teacher.UsuarioID !== UsuarioID);
      },
      error: (error: any) => {
        console.error('Error al eliminar el docente:', error);
        alert('Hubo un error al intentar eliminar el docente. Por favor, intenta de nuevo.');
```

```

ngOnInit() {
  this.cargarCuestionarios();
}

Complexity is 9 It's time to do something...
cargarCuestionarios() {
  Complexity is 8 It's time to do something...
  this.cuestionarioService.getQuizVersions().subscribe((data: any) => {
    Complexity is 6 It's time to do something...
    this.cuestionarios = data.cuestionarios.map((cuestionario: any) => ({
      ...cuestionario,
      fecha_publicacion:
        cuestionario.fechas && cuestionario.fechas.length > 0
        ? cuestionario.fechas[0].FechaPublicacion
        : null,
      fecha_cierre:
        cuestionario.fechas && cuestionario.fechas.length > 0
        ? cuestionario.fechas[0].FechaCierre
        : null,
    }));
  });
}

verDetallesCuestionario(cuestionarioID: number) {
  this.router.navigate(['administrador/quiz/${cuestionarioID}/detalles']);
}

irACrearCuestionario() {
  this.router.navigate(['administrador/create-quiz']);
}
}

```

Anexo 136. Funcionalidades de la lista de cuestionarios.

Para la funcionalidad de ver los detalles del cuestionario para obtener la información de los cuestionarios se creó la función “cargarCuestionario” a partir del servicio “getPreguntasRespuestas” llamando el id del cuestionario y almacenarlo en una variable data, posteriormente se agregó una función para obtener 10 preguntas por página para la función de paginado.


```

33 Complexity is 3 Everything is cool!
34 cargarCuestionario(cuestionarioID: number): void {
35     this.cuestionarioService.getPreguntasRespuestas(cuestionarioID).subscribe({
36         next: (data) => {
37             this.cuestionario = data;
38             this.preguntas = data.preguntas;
39             this.totalPaginas = Math.ceil(this.preguntas.length / this.preguntasPorPagina);
40         },
41         error: (err) => {
42             console.error('Error al cargar el cuestionario:', err);
43             alert('No se pudo cargar el cuestionario.');
```

Anexo 137. Funcionalidades para la vista detalles del cuestionario.

Para crear la nueva versión del cuestionario se agregaron funciones para poder realizar la funcionalidad. Se creó la función “cargarCuestionarioOriginal” el cual a partir de obtener el id del cuestionario con sservicio “getPreduntasRespuestas” se obtiene la información del cuestionario original y lo almacena en una variable data para que solamente obtenga las preguntas, variables y signos.

```

ngOnInit(): void {
    this.cargarCuestionarioOriginal();
}

Complexity is 5 Everything is cool!
cargarCuestionarioOriginal(): void {
    this.cuestionarioService.getPreguntasRespuestas(this.cuestionarioID).subscribe({
        Complexity is 3 Everything is cool!
        next: (data) => {
            this.nombre = ''; // Nombre vacío por defecto
            this.descripcion = ''; // Descripción vacía por defecto
            this.preguntas = data.preguntas.map((pregunta: any) => ({
                ...pregunta,
                Texto: pregunta.Texto, // Copiar el texto original de la pregunta
                Tipo: pregunta.Tipo, // Copiar el tipo original de la pregunta
            }));
        },
        error: (err) => {
            console.error('Error al cargar el cuestionario original:', err);
            alert('Hubo un error al cargar el cuestionario original.');
```

Anexo 138. Función “cargarCuestionarioOriginal”

Para la función de crear nuevo cuestionario se creó la función “crearCuestionario” en ella se asigna el nombre y descripción de forma predeterminada o por el usuario, luego de asignar las preguntas y el signo de la pregunta se procede a mandar el nuevo cuestionario al backend mediante el servicio “createQuiz” de “cuestionarioService”. Posteriormente muestra los mensajes si el cuestionario se ha creado exitosamente o Error al crear cuestionario.

```
Complexity is 8 It's time to do something...
crearCuestionario(): void {
  const nuevoCuestionario = {
    nombre: this.nombre || 'Nuevo Cuestionario',
    descripcion: this.descripcion || 'Sin descripción',
    preguntas: this.preguntas.map((pregunta) => ({
      Texto: pregunta.Texto,
      Tipo: pregunta.Tipo,
      variables: pregunta.variables.map((variable: any) => variable.VariableID),
    })),
  };

  // Enviar el nuevo cuestionario al backend
  this.cuestionarioService.createQuiz(nuevoCuestionario).subscribe({
    next: () => {
      alert('Cuestionario creado exitosamente');
      this.router.navigate(['/quiz']);
    },
    error: (err) => {
      console.error('Error al crear el cuestionario:', err);
      alert('Hubo un error al crear el cuestionario.');
```

Anexo 139. Función “crearCuestionario”

```

1 <div class="crear-cuestionario">
2   <form (ngSubmit)="crearCuestionario()">
3     <h3>CREAR UN NUEVO CUESTIONARIO</h3>
4     <label>Nombre</label>
5     <input type="text" [(ngModel)]="nombre" name="nombre" required />
6     <label>Descripción</label>
7     <textarea [(ngModel)]="descripcion" name="descripcion"></textarea>
8     <h3>PREGUNTAS DEL CUESTIONARIO</h3>
9     <div *ngFor="let pregunta of obtenerPreguntasPagina(); let i = index" class="pregunta-row">
10      <div class="pregunta-col">
11        <label>Pregunta {{ (paginaActual - 1) * preguntasPorPagina + (i + 1) }}</label>
12        <input
13          type="text"
14          placeholder="Texto de la pregunta"
15          [(ngModel)]="pregunta.Texto"
16          name="textoPregunta{{ i }}"
17          required
18        />
19      </div>
20      <div class="variable-col">
21        <label>Variables</label>
22        <div class="variable-container">
23          <div *ngFor="let variable of pregunta.variables" class="variable-item">
24            {{ variable.Nombre }}
25          </div>
26        </div>
27      </div>
28      <div class="tipo-col">
29        <label>Signo</label>
30        <select [(ngModel)]="pregunta.Tipo" name="tipoPregunta{{ i }}">
31          <option value="Positiva">Positiva</option>
32          <option value="Negativa">Negativa</option>
33        </select>
34      </div>
35    </div>
36  </div>

```

Anexo 140. Código de la vista del formulario para crear nueva versión del cuestionario.

Models

Los modelos de Angular creados para este sistema están diseñados para reflejar la estructura de los datos que se manejan en el backend. Cada modelo representa una entidad del sistema, como Alumno, Grupo, Cuestionario, y Pregunta, entre otros, y se encargan de gestionar la información relevante para cada componente. Estos modelos permiten que los datos se puedan manipular fácilmente dentro de la aplicación, asegurando una comunicación fluida entre el frontend y el backend mediante servicios HTTP.

```

You, el mes pasado | 1 author (You)
1  export interface ResponseGet<T> {
2      Institucion: any;
3      data: T;
4      status: number;
5      message: string;
6  }
7
You, hace 3 meses | 1 author (You)
8  export interface ResponsePPD {
9      status: number;
10     message: string;
11 }
12
You, hace 3 meses | 1 author (You)
13 export interface ErrorExcep {
14     status: number;
15     message: string;
16 }

```

Anexo 141. Modelo Responses

Para el modelo usuarios se añadieron las interfaces para la estructura de los datos del estudiante, maestro y administrador para diferentes funciones, que en este caso son para crear, obtener y actualizar.

```

1  import { Countries, Degrees, Institutions, } from "../catalogs";
2  import { Grupo} from "../gruops";
3
You, hace 3 meses | 1 author (You)
4  export interface createStudent{
5      Nombre: string,
6      Primer_apellido: string,
7      Segundo_apellido: string,
8      Matricula: string,
9      Semestre: string,
10     Tipo: string,
11     Correo: string,
12     Password: string,
13     edad: string,
14     genero: string,
15     InstitucionID: Institutions,
16     IDPais: Countries,
17     CarreraID: Degrees
18 }
19
You, hace 3 meses | 1 author (You)
20 export interface createTeacher{
21     Nombre: string,
22     Primer_apellido: string,
23     Segundo_apellido: string,
24     Matricula: string,
25     Semestre: null,
26     Tipo: string,
27     Correo: string,
28     Password: string,
29     edad: string,
30     genero: string,
31     InstitucionID: Institutions,
32     IDPais: Countries,
33     CarreraID: null,
34 }

```

Anexo 142. Modelo usuarios parte 1

You, hace 3 meses | 1 author (You)

```
export interface createAdmi{
  Nombre: string,
  Primer_apellido: string,
  Segundo_apellido: string,
  Matricula: string,
  Semestre: null,
  Tipo: string,
  Correo: string,
  Password: string,
  edad: string,
  genero: string,
  InstitucionID: Institutions,
  IDPais: Countries,
  CarreraID: null,
}
```

You, el mes pasado | 1 author (You)

```
export interface UpdateAdmi {
  Nombre?: string;
  Primer_apellido?: string;
  Segundo_apellido?: string;
  Matricula?: string;
  Semestre?: string | null;
  Tipo?: string;
  Correo?: string;
  Password?: string;
  edad?: string;
  genero?: string;
  InstitucionID?: number;
  IDPais?: number;
  CarreraID?: number | null;
}
```

Anexo 143. Modelo usuarios parte 2.

```

71 export interface getStudents{
72   UsuarioID: number;
73   Nombre: string;
74   Primer_apellido: string;
75   Segundo_apellido: string;
76   Matricula: string;
77   Semestre: string;
78   Tipo: string;
79   Correo: string;
80   Password: string;
81   edad: string;
82   genero: string;
83   InstitucionID: Institutions;
84   IDPais: Countries;
85   CarreraID: Degrees;
86   created_at: string;
87   updated_at: string;
88   grupos: Grupo;
89 }
90
91
92
93 export interface getTeachers{
94   UsuarioID: number,
95   Nombre: string,
96   Primer_apellido: string,
97   Segundo_apellido: string,
98   Matricula: string,
99   Semestre: null,
100   Tipo: string,
101   Correo: string,
102   Password: string,
103   edad: string,
104   genero: string,
105   InstitucionID: Institutions,
106   IDPais: Countries,
107   CarreraID: null,
108 }

```

Anexo 144. Modelo usuarios parte 3.

El modelo catálogos contiene las interfaces para los datos de las instituciones, países, crear países, carreras y generaciones.

```

1  export interface Institutions{
2      InstitucionID: number;
3      Nombre: string;
4      IDPais: number;
5      created_at?: string;
6      updated_at?: string;
7  }
8
9  You, hace 3 meses | 1 author (You)
10 export interface Countries {
11     IDPais: number;
12     created_at?: string;
13     Nombre: string;
14     updated_at?: string;
15 }
16 You, el mes pasado | 1 author (You)
17 export interface CreateCountries {
18     IDPais: number;
19     created_at?: string;
20     Nombre: string;
21     updated_at?: string;
22 }

```

Anexo 145. Modelo catálogos parte 1.

```

23 export interface Degrees{
24     CarreraID: number;
25     Nombre: string;
26     InstitucionID: number;
27     created_at?: string;
28     updated_at?: string;
29 }
30
31 You, el mes pasado | 2 authors (Fernanda-Chable and one other)
32 export interface Generacion {
33     GeneracionID: number;
34     Nombre: string;
35     CarreraID: number;
36     created_at?: string;
37     updated_at?: string;
38 }

```

Anexo 146. Modelo catálogos parte 2.

Para el modelo cuestionarios se creó para obtener la estructura de las respuestas del backend.

```

Fernanda-Chable, el mes pasado | 1 author (Fernanda-Chable)
1 interface CuestionarioPendiente {
2   CuestionarioID: number;
3   Nombre: string;
4   Descripcion: string;
5   FechaPublicacion: string;
6   FechaCierre: string;
7 }
8
Fernanda-Chable, el mes pasado | 1 author (Fernanda-Chable)
9 interface GrupoConCuestionarios {
10  nombreGrupo: string;
11  cuestionarios: CuestionarioPendiente[];
12 }
13
Fernanda-Chable, el mes pasado | 1 author (Fernanda-Chable)
14 export interface Cuestionario {
15   CuestionarioID: number;
16   GrupoID: number;
17   Nombre: string;
18   Descripcion: string;
19   FechaPublicacion: string;
20   FechaCierre: string;
21   created_at: string;
22   updated_at: string;
23 }

```

Anexo 147. Modelo Cuestionarios.

Para el modelo grupos se crearon las interfaces para obtener las estructuras para los grupos junto con los alumnos.

```

< export interface Grupo {
  GrupoID: number;
  Nombre: string;
  codigo_acceso: string;
  semestre: string;
  GeneracionID: Generacion;
  CarreraID: Degrees;
  created_at: string;
  updated_at: string;
  pivot: {
    UsuarioID: number;
    GrupoID: number;
    CarreraID: number;
  };
  carrera: Carrera;
}

You, hace 1 minuto | 2 authors (Fernanda-Chable and one other)
< export interface Alumno {
  UsuarioID: number;
  Nombre: string;
  Primer_apellido: string;
  Segundo_apellido: string;
  Matricula: string;
  semestre: string;
  Tipo: string;
  Correo: string;
  Password: string;
  edad: string;
  genero: string;
  InstitucionID: Institutions;
  IDPais: Countries;
  CarreraID: Degrees;
  created_at: string;
  updated_at: string;
  grupos: Grupo[];
}

```

Anexo 148. Modelo grupos.

Services

Los servicios en Angular que hemos creado son responsables de manejar la lógica de negocio y la comunicación con el backend mediante solicitudes HTTP. Estos servicios permiten obtener y enviar datos de manera eficiente, interactuando con rutas API para obtener información como cuestionarios, grupos, usuarios y puntajes. Además, se encargan de gestionar la autenticación y la sesión del usuario. Al centralizar las operaciones de datos, los servicios facilitan la reutilización de código y la mantenibilidad del sistema, asegurando una interacción fluida entre los componentes y el servidor.

```
8   export class AuthService {
9     private apiUrl = 'http://127.0.0.1:8000';
10
11     constructor(private http: HttpClient) {}
12
13     login(data: any): Observable<any> {
14       return this.http.post(`${this.apiUrl}/login`, data);
15     }
16
17     logout(): void {
18       localStorage.removeItem('access_token');
19       localStorage.removeItem('user_id');
20     }
21
22     isLoggedIn(): boolean {
23       return !!localStorage.getItem('access_token');
24     }
25
26     getToken(): string | null {
27       return localStorage.getItem('access_token');
28     }
29     setUserId(id: number): void {
30       localStorage.setItem('user_id', id.toString());
31     }
32
33     getUserId(): number | null {
34       const userId = localStorage.getItem('user_id');
35       return userId ? parseInt(userId) : null;
36     }
37   }
```

Anexo 149. Servicio Autenticación

Para el manejo de los usuarios se creó el servicio el cual tiene las funciones para registrar a los usuarios, obtener la lista de los usuarios, crearlos y de igual manera eliminarlos.

```
export class UserService {
  private URL = 'http://127.0.0.1:8000/usuarios';

  constructor(private http: HttpClient) { }

  registerStudent(user: createStudent): Observable<ResponsePPD> {
    return this.http.post<ResponsePPD>(this.URL + '/register', user);
  }

  getStudents(): Observable<ResponseGet<getStudents[]>>{
    return this.http.get<ResponseGet<getStudents[]>>(this.URL + '/alumnos');
  }

  registerTeacher(user: createTeacher): Observable<ResponsePPD> {
    return this.http.post<ResponsePPD>(this.URL + '/register', user);
  }

  getTeachers(): Observable<ResponseGet<getTeachers[]>>{
    return this.http.get<ResponseGet<getTeachers[]>>(this.URL + '/maestros');
  }

  // Método para obtener datos de un maestro específico por su ID
  getTeacher(id: number): Observable<ResponseGet<getTeachers>>{
    return this.http.get<ResponseGet<getTeachers>>(`${this.URL}/maestro/${id}`);
  }
}
```

Anexo 150. “userService” parte 1.

```
39 registerAdmin(user: createAdmi): Observable<ResponsePPD> {
40   return this.http.post<ResponsePPD>(this.URL + '/register', user);
41 }
42
43 getAdmins(): Observable<ResponseGet<getAdmins[]>>{
44   return this.http.get<ResponseGet<getAdmins[]>>(this.URL + '/administradores');
45 }
46
47 // Método para obtener datos de un alumno específico por su ID
48 getStudent(id: number): Observable<ResponseGet<getStudents>> {
49   return this.http.get<ResponseGet<getStudents>>(`${this.URL}/alumnos/${id}`);
50 }
51
52 getAdministrator(id: number): Observable<ResponseGet<getAdmins>> {
53   return this.http.get<ResponseGet<getAdmins>>(`${this.URL}/administrador/${id}`);
54 }
55
56 //Metodo para obtener alumnos por institucionID
57 getStudentsInst(InstitutionID: string): Observable<ResponseGet<getStudents[]>>{
58   return this.http.get<ResponseGet<getStudents[]>>(`${this.URL}/alumnos/institucion/${InstitutionID}`)
59 }
60
```

Anexo 151. “userService” parte 2.

```

61  ✓ getTeachersInst(InstitutionID: string): Observable<ResponseGet<getTeachers[]>>{
62      return this.http.get<ResponseGet<getTeachers[]>>(`${this.URL}/maestros/institucion/${InstitutionID}`);
63  }
64
65  // Método para actualizar el perfil de administrador
66  ✓ updateAdmin(UsuarioID: string, data: UpdateAdml): Observable<ResponsePPD> {
67      return this.http.put<ResponsePPD>(`${this.URL}/actualizar/${UsuarioID}`, data);
68  }
69
70  ✓ deleteStudent(UsuarioID: number): Observable<ResponsePPD> {
71      return this.http.delete<ResponsePPD>(`${this.URL}/eliminar/${UsuarioID}`);
72  }
73
74  ✓ deleteTeacher(UsuarioID: number): Observable<ResponsePPD> {
75      return this.http.delete<ResponsePPD>(`${this.URL}/eliminar/${UsuarioID}`);
76  }
77  }

```

Anexo 152. “userService” parte 3.

Para obtener todos los catálogos se realizó el servicio catalogs este contiene los servicios para obtener las instituciones, países, generaciones y carreras, de igual manera para crear un nuevo país.

```

10  export class CatalogsService {
11      private URL = "http://127.0.0.1:8000/catalogs";
12
13      constructor(private http: HttpClient) {}
14
15      getInstitutionsId(IDPais: string): Observable<ResponseGet<Institutions[]>> {
16          return this.http.get<ResponseGet<Institutions[]>>(`${this.URL}/instituciones?IDPais=${IDPais}`);
17      }
18
19      getInstitutions(): Observable<ResponseGet<Institutions[]>>{
20          return this.http.get<ResponseGet<Institutions[]>>(this.URL + "/ainstituciones");
21      }
22
23      getInstitution(InstitutionID: string): Observable<ResponseGet<Institutions[]>> {
24          return this.http.get<ResponseGet<Institutions[]>>(`${this.URL}/idinstituciones/${InstitutionID}`);
25      }
26
27      getCountries(): Observable<ResponseGet<Countries[]>>{
28          return this.http.get<ResponseGet<Countries[]>>(this.URL + "/países");
29      }
30  }

```

Anexo 153. “catalogsService” parte 1.

```

31  getDegreesId(InstitucionID: string): Observable<ResponseGet<Degrees[]>>{
32      return this.http.get<ResponseGet<Degrees[]>>(`${this.URL}/carreras?InstitucionID=${InstitucionID}`);
33  }
34
35  getDegrees(): Observable<ResponseGet<Degrees[]>>{
36      return this.http.get<ResponseGet<Degrees[]>>(this.URL + "/acarreras")
37  }
38
39  getGeneraciones(): Observable<ResponseGet<Generacion[]>> {
40      return this.http.get<ResponseGet<Generacion[]>>(`${this.URL}/generaciones`);
41  }
42
43  createCountry(country: CreateCountries): Observable<ResponsePPD> {
44      return this.http.post<ResponsePPD>(this.URL + '/agregarpais', country);
45  }
46
47  }

```

Anexo 154. “catalogsService” parte 2.

Para obtener el alumno por grupo se creó el siguiente servicio.

```
9 export class AlumnoGruposService {
10   private apiUrl = 'http://127.0.0.1:8000/alumno';
11
12   constructor(private http: HttpClient) {}
13
14   // Complexity is 3 Everything is cool!
15   getAlumnoGrupos(usuarioID: number): Observable<{ data: Grupo[], status: number, message: string }> {
16     const url = `${this.apiUrl}/${usuarioID}/grupos`;
17     const headers = new HttpHeaders({
18       'Authorization': `Bearer ${localStorage.getItem('access_token') || ''}`
19     });
20     return this.http.get<{ data: Grupo[], status: number, message: string }>(url, { headers });
21   }
22 }
```

Anexo 155. “AlumnosGruposService”.

Para el servicio de los cuestionarios se crearon las funciones para obtener los cuestionarios, preguntas, versiones de los cuestionarios, publicar cuestionarios, etc.

```
8 export class CuestionarioService {
9   private apiUrl = 'http://127.0.0.1:8000';
10
11   constructor(private http: HttpClient) {}
12
13   // Complexity is 4 Everything is cool!
14   getPreguntasRespuestas(cuestionarioID: number): Observable<any> {
15     return this.http.get<any>(`${this.apiUrl}/cuestionarios/${cuestionarioID}/preguntas`);
16   }
17
18   // Complexity is 5 Everything is cool!
19   enviarRespuestas(usuarioID: number, respuestas: any): Observable<any> {
20     const payload = { respuestas };
21     return this.http.post<any>(`${this.apiUrl}/respuestas/${usuarioID}/guardarRespuestas`, payload, {
22       headers: { 'Content-Type': 'application/json' }
23     });
24   }
25
26   // Obtener todas las preguntas
27   // Complexity is 3 Everything is cool!
28   getAllQuestions(): Observable<any> {
29     return this.http.get(`${this.apiUrl}/preguntas`);
30   }
31 }
```

Anexo 156. “cuestionarioService” parte 1.

```
28 // Crear un nuevo cuestionario
29 // Complexity is 4 Everything is cool!
30 createQuiz(data: any): Observable<any> {
31   return this.http.post(`${this.apiUrl}/cuestionarios`, data);
32 }
33
34 // Complexity is 4 Everything is cool!
35 getQuizVersions(): Observable<any> {
36   return this.http.get<any>(`${this.apiUrl}/cuestionarios`);
37 }
38
39 // Complexity is 5 Everything is cool!
40 publishQuiz(cuestionarioID: number, payload: any): Observable<any> {
41   console.log('Payload enviado:', payload); // Para verificar Los datos enviados
42   return this.http.post<any>(`${this.apiUrl}/cuestionarios/${cuestionarioID}/publicar`, payload);
43 }
44
45 // Método para obtener los cuestionarios aplicados por el usuario (maestro)
46 // Complexity is 4 Everything is cool!
47 obtenerCuestionariosPorUsuario(usuarioID: number): Observable<any> {
48   return this.http.get<any>(`${this.apiUrl}/cuestionarios/usuario/${usuarioID}`);
49 }
50
51 // Método para obtener todos los cuestionarios
52 // Complexity is 3 Everything is cool!
53 getAllCuestionarios(): Observable<any> {
54   return this.http.get(`${this.apiUrl}/cuestionarios/versions`);
55 }
56 }
```

Anexo 157. “cuestionarioService” parte 2.

```

Complexity is 4 Everything is cool!
50  getAllGroups(): Observable<any> {
51      return this.http.get<any>(`${this.apiUrl}/grupos`);
52  }
53
Complexity is 4 Everything is cool!
54  saveNewQuizVersion(quizId: number, data: any): Observable<any> {
55      return this.http.post(`${this.apiUrl}/cuestionarios/${quizId}/duplicar-editar`, data);
56  }
57
Complexity is 5 Everything is cool!
58  duplicarCuestionario(cuestionarioID: number, payload: any): Observable<any> {
59      return this.http.post<any>(`${this.apiUrl}/cuestionarios/${cuestionarioID}/duplicar`, payload);
60  }
61
62  }
63

```

Anexo 158. “cuestionarioService” parte 3.

Servicio para obtener puntajes de las variables del cuestionario.

```

8  export class PuntajeService {
9      private apiUrl = 'http://127.0.0.1:8000/puntajes';
10
11      constructor(private http: HttpClient) { }
12
13      obtenerPuntajesVariables(usuarioID: number): Observable<any> {
14          return this.http.get<any>(`${this.apiUrl}/${usuarioID}`);
15      }
16
17  }
18

```

Anexo 159. “PuntajeService”.

Servicio para obtener los puntajes por el cuestionario.

```

8  export class ResultsService {
9      private apiUrl = 'http://127.0.0.1:8000';
10
11      constructor(private http: HttpClient) { }
12
13      obtenerPuntajesPorCuestionario(usuarioID: number, cuestionarioID: number): Observable<any> {
14          return this.http.get(`${this.apiUrl}/puntajesporCuestionario/${usuarioID}/${cuestionarioID}`);
15      }
16
17  }
18

```

Anexo 160. “PuntajeService”.

Servicio para obtener los cuestionarios resueltos.

```
8   export class SolvedQuizzesService {
9     private apiUrl = 'http://127.0.0.1:8000';
10
11    constructor(private http: HttpClient) { }
12
13    // Marcar un cuestionario como resuelto
14    // Complexity is 4 Everything is cool!
15    marcarCuestionarioResuelto(data: any): Observable<any> {
16      return this.http.post(`${this.apiUrl}/cuestionarios/marcar-resuelto`, data);
17    }
18
19    // Obtener cuestionarios resueltos por usuario y grupo
20    // Complexity is 3 Everything is cool!
21    obtenerCuestionariosResueltos(usuarioId: number, groupId: number): Observable<any> {
22      return this.http.get(`${this.apiUrl}/cuestionarios/resueltos/${usuarioId}/${groupId}`);
23    }
24
25    // Complexity is 4 Everything is cool!
26    getCuestionariosResueltosPorAlumno(alumnoId: number): Observable<any> {
27      return this.http.get<any>(`${this.apiUrl}/cuestionarios-resueltos/alumno/${alumnoId}`);
28    }
29
30    // Complexity is 3 Everything is cool!
31    obtenerCuestionariosPendientes(usuarioId: number, groupId: number): Observable<any> {
32      return this.http.get(`${this.apiUrl}/cuestionarios/pendientes/${usuarioId}/${groupId}`);
33    }
34  }
```

Anexo 161. “SolvedQuizzesService”.

Servicio para obtener los grupos del maestro.

```
11  export class TeacherGroupsService {
12    private apiUrl = 'http://127.0.0.1:8000';
13
14    constructor(private http: HttpClient) { }
15
16    // Complexity is 4 Everything is cool!
17    getGruposDeMaestro(usuarioId: number): Observable<any> {
18      return this.http.get<any>(`${this.apiUrl}/maestro/${usuarioId}/grupos`);
19    }
20
21    getContadorGruposDeMaestro(usuarioId: number): Observable<Grupo[]> {
22      return this.http.get<Grupo[]>(`${this.apiUrl}/maestro/${usuarioId}/grupos`);
23    }
24
25    // metodo para obtener los cuestionarios aplicados por grupo
26    getAppliedQuizzesByGroup(groupId: number): Observable<{data: Cuestionario[]}> {
27      const url = `${this.apiUrl}/cuestionarios-aplicados/grupo/${groupId}`;
28      return this.http.get<{data: Cuestionario[]}>(url);
29    }
30
31    // Nuevo método para obtener la información de un grupo específico
32    getGroupById(groupId: number): Observable<{ data: Grupo }> {
33      const url = `${this.apiUrl}/grupos/${groupId}`;
34      return this.http.get<{ data: Grupo }>(url);
35    }
36  }
```

Anexo 162. “TeacherGroupService” parte 1.

```

36 // Método para crear un nuevo grupo
    Complexity is 5 Everything is cool!
37 createGroup(grupoData: any): Observable<any> {
38     const url = `${this.apiUrl}/crear-grupos`;
39     return this.http.post<any>(url, grupoData);
40 }
41 // Método para obtener los integrantes de un grupo
    Complexity is 4 Everything is cool!
42 getIntegrantes(grupoId: number): Observable<any> {
43     return this.http.get<any>(`${this.apiUrl}/grupos/${grupoId}/integrantes`);
44 }
45
46 // Método para buscar alumnos
    Complexity is 3 Everything is cool!
47 buscarAlumno(nombre: string): Observable<any> {
48     const params = new HttpParams().set('nombre', nombre);
49     return this.http.get(`${this.apiUrl}/alumno/buscar`, { params });
50 }
51
52 // Método para añadir alumno al grupo
    Complexity is 3 Everything is cool!
53 agregarAlumnoAGrupo(grupoId: number, alumnoId: number): Observable<any> {
54     return this.http.post(`${this.apiUrl}/grupos/${grupoId}/alumnos/agregar`, { UsuarioID: alumnoId });
55 }
56 }

```

Anexo 163. “TeacherGroupService” parte 2.

Validators

Los validadores en Angular que hemos implementado se encargan de asegurar que los datos ingresados por los usuarios sean correctos antes de ser enviados al servidor. Estos validadores verifican que los campos del formulario cumplan con requisitos como ser obligatorios, tener un formato válido (como correos electrónicos o contraseñas) o estar dentro de un rango específico. Al centralizar esta lógica de validación, mejoramos la experiencia del usuario, garantizando que solo se procesen datos válidos y evitando errores en la interacción con la base de datos.

Validador de confirmación de contraseña

```

ProyectoCEP > src > app > core > validators > confirmPassword.validator.ts > ...
1  import { AbstractControl, FormGroup, ValidationErrors, ValidatorFn } from '@angular/forms';
2
3  export const confirmPassword: ValidatorFn = (control: AbstractControl): ValidationErrors | null => {
4      const formGroup = control as FormGroup;
5      const password = formGroup.get('contraseña')?.value;
6      const confirmPassword = formGroup.get('confirmcontraseña')?.value;
7
8      if (password === confirmPassword) {
9          return null;
10     }
11
12     return {
13         passwordMismatch: {
14             message: 'Las contraseñas no coinciden',
15         },
16     };
17 };
18

```

Anexo 164. Validador de confirmación de contraseña

Email validado

```

ProyectoCEP > src > app > core > validators > email.validator.ts > ...
1  import { AbstractControl, ValidatorFn } from '@angular/forms';
2  import { InputValidationResponse } from './InputValidationResponse';
3
4  export function emailValidator(): ValidatorFn {
5    return function validator({ value }: AbstractControl): InputValidationResponse {
6
7      if (value === null || value === undefined || value === '') {
8        return {
9          message: 'Campo requerido',
10        };
11      }
12
13      const VALID_EMAIL_REGEX = new RegExp(/^[a-zA-Z0-9ñÑ._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/);
14
15      if (VALID_EMAIL_REGEX.test(value)) {
16        return null;
17      }
18
19      return {
20        message: 'Formato incorrecto',
21      };
22    };
23  }

```

Anexo 165. Email validado

Validador de longitud

```

ProyectoCEP > src > app > core > validators > length.validator.ts > ...
1  import { AbstractControl, ValidatorFn } from '@angular/forms';
2  import { InputValidationResponse } from './InputValidationResponse';
3
4  interface Length {
5    min: number;
6    max: number;
7  }
8
9  export function lengthValidator({ min, max, optional = false }: { min: number; max: number; optional?: boolean }): ValidatorFn {
10    return function validator({
11      value,
12    }: AbstractControl): InputValidationResponse {
13      if (optional && (value === null || value === undefined || value === '')) {
14        return null;
15      }
16
17      if (value.length === 0 && optional == false) {
18        return {
19          message: `Campo requerido`,
20        };
21      }
22
23      if (value.length < min) {
24        return {
25          message: `La longitud mínima debe ser de ${min}`,
26        };
27      }
28
29      if (value.length > max) {
30        return {
31          message: `La longitud máxima debe ser de ${max}`,
32        };
33      }
34
35      return null;
36    };
37  }

```

Anexo 166. Validador de longitud

Validador de campo requerido

```
ProyectoCEP > src > app > core > validators >  makeRequired.validator.ts > ...  
1  import { AbstractControl } from "@angular/forms";  
2  import { InputValidationResponse } from "../validators/InputValidationResponse";  
3  
4  export function makeRequired(control: AbstractControl): InputValidationResponse {  
5      if(control.value?.length == 0) {  
6          return {  
7              message: 'Campo requerido'  
8          }  
9      }  
10  
11      return null;  
12  }  
13
```

Anexo 167. Validador de campo requerido

Validador de nombre

```
ProyectoCEP > src > app > core > validators >  nameValidator.ts > ...  
● 1  import { InputValidationResponse } from './InputValidationResponse';  
2  import { AbstractControl } from '@angular/forms';  
3  
4  export function nameValidator( { value }: AbstractControl  
5  ): InputValidationResponse {  
6      const VALID_NAME_REGEX = new RegExp(/^(?:\p{Lu}[\p{L}'-]*\s?)+$/u);  
7  
8      if (VALID_NAME_REGEX.test(value)) {  
9          return null;  
10     }  
11  
12     return {  
13         message: 'Formato incorrecto',  
14     };  
15 }  
16
```

Anexo 168. Validador de nombre

Validadora contraseña confirmada

```

ProyectoCEP > src > app > core > validators > password.validator.ts > ...
1  import { AbstractControl, ValidationErrors, ValidatorFn } from '@angular/forms';
2
3  export const passwordsMatchValidator: ValidatorFn = (control: AbstractControl): ValidationErrors | null => {
4    const formGroup = control as any;
5    const password = formGroup.get('password')?.value;
6    const confirmPassword = formGroup.get('confirmpassword')?.value;
7
8    if (password && confirmPassword && password !== confirmPassword) {
9      return { passwordsDoNotMatch: true };
10   }
11   return null;
12 };
13

```

Anexo 169. Validadora contraseña confirmada

Validador de verificación de matrícula o correo

```

ProyectoCEP > src > app > core > validators > serverError.validator.ts > ...
1  import { AbstractControl } from '@angular/forms';
2
3  export class ServerErrorValidator {
4    static setServerErrors(errors: { [key: string]: string[] }, form: AbstractControl): void {
5      for (const field in errors) {
6        if (errors.hasOwnProperty(field)) {
7          const control = form.get(field.toLowerCase());
8          if (control) {
9            const translatedMessages = errors[field].map((msg: string) => {
10              if (field === 'Matricula' && msg === 'The matricula has already been taken.') {
11                return 'La matricula ya ha sido registrada.';
12              }
13              if (field === 'Correo' && msg === 'The correo has already been taken.') {
14                return 'El correo ya ha sido registrado.';
15              }
16              return msg;
17            });
18            control.setErrors({ serverError: translatedMessages.join(', ') });
19          }
20        }
21      }
22    }
23  }
24

```

Anexo 170. Validador de verificación de matrícula o correo

Routes

Los routers en Angular que hemos creado gestionan la navegación entre diferentes vistas y componentes del sistema. Cada ruta está configurada para asociar una URL específica con un componente, permitiendo que los usuarios interactúen con distintas secciones del sistema de forma dinámica. Esto facilita la carga eficiente de los componentes sin tener que recargar toda la página, mejorando la experiencia del usuario. Además, los routers también manejan la autenticación y autorización, asegurando que los usuarios solo puedan acceder a las rutas correspondientes según sus permisos.

Routers página principal

```

3 import { Routes } from '@angular/router';
4 import { AuthGuard } from '../auth.guard';
5 import { ListstudentGroupComponent } from './administrator/institutions/liststudent-group/liststudent-group.component';
6 import { AdmUsuariosComponent } from './administrator/usuarios/adm-usuarios/adm-usuarios.component';
7 import { AddCountryComponent } from './administrator/countries/add-country/add-country.component';
8 import { AdmCountriesComponent } from './administrator/countries/adm-countries/adm-countries.component';
9 import { CountryInstitutionsComponent } from './administrator/countries/country-institutions/country-institutions.component';
10 import { QuizManagementComponent } from './administrator/quiz/quiz-management/quiz-management.component';
11 import { QuizDetailsComponent } from './student/quiz-details/quiz-details/quiz-details.component';
12
13 export const routes: Routes = [
14   {
15     path: '',
16     loadComponent: () =>
17       import('../layout/inicio/inicio.component').then((c) => c.InicioComponent),
18     children: [
19       {
20         path: '',
21         loadComponent: () =>
22           import('./home/home/home.component').then((c) => c.HomeComponent)
23       },
24       {
25         path: 'home',
26         loadComponent: () =>
27           import('./home/home/home.component').then((c) => c.HomeComponent)
28       },
29       {
30         path: 'locacion',
31         loadComponent: () =>
32           import('./locacion/locacion.component').then((c) => c.LocacionComponent)
33       },
34       {
35         path: 'quien_somos',
36         loadComponent: () =>
37           import('./quienes_somos/quienes-somos/quienes-somos.component').then((c) => c.QuienesSomosComponent)
38       },
39     ],
40   },
41 ];

```

Anexo 171. Routers página principal

```

ProyectoCEP > src > app > ui > app.routes.ts > ...
13 export const routes: Routes = [
18   children: [
39     {
40       path: 'bloqueo',
41       loadComponent: () =>
42         import('./blockage/blockage/blockage.component').then((c) => c.BlockageComponent)
43     },
44     {
45       path: 'cuestionario',
46       loadComponent: () =>
47         import('./cuestionario/cuestionario/cuestionario.component').then((c) => c.CuestionarioComponent),
48       canActivate: [AuthGuard]
49     },
50     {
51       path: 'instrucciones',
52       loadComponent: () =>
53         import('./cuestionario/instrucciones/instrucciones.component').then((c) => c.InstruccionesComponent),
54       canActivate: [AuthGuard]
55     },
56     {
57       path: 'puntaje',
58       loadComponent: () =>
59         import('./cuestionario/puntaje/puntaje.component').then((c) => c.PuntajeComponent),
60       canActivate: [AuthGuard]
61     },
62     {
63       path: 'registro',
64       title: 'Registrar estudiante',
65       loadComponent: () =>
66         import('./student/student-register/student-register.component').then((c) => c.studentRegisterComponent),
67     },
68   ],
69 ];

```

Anexo 172. Routers página principal (Parte 2)

```

    import('./student/student-register/student-register.component').then((c) => c.studentRegisterComponent),
  },
  {
    path: 'registro-admin',
    title: 'Registrar administrador',
    loadComponent: () =>
      import('./administrator/register-administrator/register-administrator.component').then((c) => c.RegisterAdministratorComponent),
  },
  {
    path: 'login',
    title: 'login',
    loadComponent: () =>
      import('./login/login.component').then((c) => c.LoginComponent)
  }
]
},

```

Anexo 173 Routers página principal (Parte 3)

Routes para el dashboard de alumnos

```

ProyectoCEP > src > app > ui > app.routes.ts > ...
13 export const routes: Routes = [
84   {
85     path: 'alumno',
86     loadComponent: () =>
87       import('./student/navbarEstu/navbar/navbar.component').then((c) => c.NavbarComponent),
88     children: [
89       {
90         path: '',
91         loadComponent: () =>
92           import('./student/dashboard-Estu/dashboard/dashboard.component').then((c) => c.DashboardComponent),
93       },
94       {
95         path: 'dash-estu',
96         loadComponent: () =>
97           import('./student/dashboard-Estu/dashboard/dashboard.component').then((c) => c.DashboardComponent),
98       },
99       {
100        path: 'gruops',
101        loadComponent: () =>
102          import('./student/navbarEstu/gruops-Estu/gruops/gruops.component').then((c) => c.GruopsComponent),
103      },
104      {
105        path: 'cuestionarios-resueltos',
106        loadComponent: () =>
107          import('./student/navbarEstu/gruops-Estu/solved-quizzes/solved-quizzes.component').then((c) => c.SolvedQuizzesComponent),
108      },
109      {
110        path: 'resultados/:CuestionarioID',
111        loadComponent: () =>
112          import('./student/navbarEstu/gruops-Estu/results/results.component').then((c) => c.ResultsComponent),
113      },
114      {
115        path: 'resultados/:UsuarioID/:CuestionarioID',
116        loadComponent: () =>
117          import('./student/navbarEstu/gruops-Estu/results/results.component').then((c) => c.ResultsComponent),
118      }
119    ]
120  }
121 ]

```

Anexo 174. Routes para el dashboard de alumnos (parte 1)

```

ProyectoCEP > src > app > ui > app.routes.ts > ...
13   export const routes: Routes = [
88     children: [
117       {
118       },
119       {
120         path: 'generales',
121         loadComponent: () =>
122           import('./student/navbarEstu/general-questions/general-questions.component').then((c) => c.GeneralQuestionsComponent),
123       },
124       {
125         path: 'pendiente',
126         loadComponent: () =>
127           import('./student/dashboard-Estu/quiz-pending/quiz-pending.component').then((c) => c.QuizPendingComponent),
128       },
129       {
130         path: 'Resultados generales',
131         loadComponent: () =>
132           import('./student/navbarEstu/result-questions/result-questions.component').then((c) => c.ResultQuestionsComponent),
133       },
134       {
135         path: 'cuestionarios/:id/preguntas',
136         loadComponent: () =>
137           import('./student/quiz-details/quiz-details/quiz-details.component').then((c) => c.QuizDetailsComponent),
138       },
139     ],
140   ],

```

Anexo 175 Routes para el dashboard de alumnos (Parte 2)

Routes para el dashboard de maestro

```

ProyectoCEP > src > app > ui > app.routes.ts > ...
13   export const routes: Routes = [
277   {
278     title: 'Maestro',
279     path: 'maestro',
280     loadComponent: () =>
281       import('./teacher/navbarTeacher/navbar/navbar.component').then((c) => c.NavbarComponent),
282     children: [
283       {
284         path: '',
285         loadComponent: () =>
286           import('./teacher/dashTeacher/dashboard/dashboard.component').then((c) => c.DashboardComponent),
287       },
288       {
289         path: 'dashTeacher',
290         loadComponent: () =>
291           import('./teacher/dashTeacher/dashboard/dashboard.component').then((c) => c.DashboardComponent),
292       },
293       {
294         path: 'gruopsTeacher',
295         loadComponent: () =>
296           import('./teacher/gruops/gruops/gruops.component').then((c) => c.GruopsComponent),
297       },
298       {
299         path: 'groupQuizzes',
300         loadComponent: () =>
301           import('./teacher/gruops/group-quizzes/group-quizzes.component').then((c) => c.GroupQuizzesComponent),
302       },
303       {
304         path: 'createGroup',
305         loadComponent: () =>
306           import('./teacher/gruops/create-group/create-group.component').then((c) => c.CreateGroupComponent),
307       },

```

Anexo 176 Routes para el dashboard de maestro (Parte 1)

```

ProyectoCEP > src > app > ui > app.routes.ts > ...
13   export const routes: Routes = [
282     children: [
308       {
309         path: 'grupo/:id',
310         loadComponent: () =>
311           import('./teacher/gruops/members-groups/members-groups.component').then((c) => c.MembersGroupsComponent ),
312       },
313       {
314         path: 'alumno-search/:grupoId',
315         loadComponent: () =>
316           import('./teacher/gruops/alumno-search/alumno-search.component').then((c) => c.AlumnoSearchComponent ),
317       },
318       {
319         path: 'quiz',
320         loadComponent: () =>
321           import('./teacher/quiz/quiz-management/quiz-management.component').then((c) => c.QuizManagementComponent),
322       },
323       {
324         path: 'create-quiz',
325         loadComponent: () =>
326           import('./teacher/quiz/create-quiz/create-quiz.component').then((c) => c.CreateQuizComponent),
327       },
328       {
329         path: 'post-quiz',
330         loadComponent: () =>
331           import('./teacher/quiz/post-quiz/post-quiz.component').then((c) => c.PostQuizComponent),
332       },
333       {
334         path: 'applied-quiz',
335         loadComponent: () =>
336           import('./teacher/quiz/applied-quiz/applied-quiz.component').then((c) => c.AppliedQuizComponent),
337       },

```

Anexo 177 Routes para el dashboard de maestro (Parte 2)

```

13   export const routes: Routes = [
282     children: [
339       {
340         path: 'detailsQuiz/:grupoId',
341         loadComponent: () =>
342           import('./teacher/quiz/details-quiz/details-quiz.component').then((c) => c.DetailsQuizComponent),
343       },
344       {
345         path: 'solved-quizzes/:usuarioID',
346         loadComponent: () =>
347           import('./teacher/quiz/quizzes-list/quizzes-list.component').then((c) => c.QuizzesListComponent),
348       },
349       {
350         path: 'resultados/:usuarioID/:cuestionarioID',
351         loadComponent: () =>
352           import('./teacher/quiz/results-quiz/results-quiz.component').then((c) => c.ResultsQuizComponent),
353       },
354       {
355         path: 'listAlumnos',
356         loadComponent: () =>
357           import('./teacher/list/list-alumnos/list-alumnos.component').then((c) => c.ListAlumnosComponent),
358       },
359       {
360         path: 'macro-Resultados/:usuarioID/:cuestionarioID',
361         loadComponent: () =>
362           import('./teacher/quiz/result/macro-view/macro-view.component').then((c) => c.MacroViewComponent),
363       },
364       {
365         path: 'categorias/:usuarioID/:cuestionarioID/:categoriaID',
366         loadComponent: () =>
367           import('./teacher/quiz/result/category-view/category-view.component').then((c) => c.CategoryViewComponent),
368       },
369       {
370         path: 'expresiones/:usuarioID/:cuestionarioID/:expresionID',
371         loadComponent: () =>
372           import('./teacher/quiz/result/micro-view/micro-view.component').then((c) => c.MicroViewComponent),
373       },

```

Anexo 178. Routes para el dashboard de maestro (Parte 3)

Routes para el dashboard de administrador

```
ProyectoCEP > src > app > ui > app.routes.ts > ...
13 export const routes: Routes = [
14   //
15   {
16     title: 'Administrador',
17     path: 'administrador',
18     loadComponent: () =>
19       import('./administrator/navbar-adm/navbar-adm.component').then((c) => c.NavbarAdmComponent),
20     children: [
21       {
22         path: '',
23         loadComponent: () =>
24           import('./administrator/dashboard-adm/dashboard-adm/dashboard-adm.component').then((c) => c.DashboardAdmComponent),
25       },
26       {
27         path: 'dash-admin',
28         loadComponent: () =>
29           import('./administrator/dashboard-adm/dashboard-adm/dashboard-adm.component').then((c) => c.DashboardAdmComponent),
30       },
31       {
32         path: 'per-fil/:id',
33         loadComponent: () =>
34           import('./administrator/adm-profile/adm-profile.component').then((c) => c.AdmProfileComponent),
35       },
36       {
37         path: 'registro-docente',
38         title: 'Registrar docente',
39         loadComponent: () =>
40           import('./teacher/teacher-register/teacher-register.component').then((c) => c.teacherRegisterComponent),
41       },
42       {
43         path: 'lista-estudiantes',
44         title: 'Lista de estudiantes',
45         loadComponent: () =>
46           import('./administrator/list-students/list-students.component').then((c) => c.ListStudentsComponent),
47       },
48     ],
49   },
50 ];
```

Anexo 179. Routes para el dashboard de administrador (Parte 1)

```
ProyectoCEP > src > app > ui > app.routes.ts > routes > children > loadComponent
export const routes: Routes = [
  children: [
    {
      path: 'lista-docentes',
      title: 'Lista de docentes',
      loadComponent: () =>
        import('./administrator/list-teachers/list-teachers.component').then((c) => c.ListTeachersComponent),
    },
    {
      path: 'instituciones',
      title: 'Instituciones',
      loadComponent: () =>
        import('./administrator/institutions/institutions/institutions.component').then((c) => c.InstitutionsComponent),
    },
    {
      path: 'instituciones/:id',
      title: 'Institucion',
      loadComponent: () =>
        import('./administrator/institutions/details-institution/details-institution.component').then((c) => c.DetailsInstitutionComponent),
    },
    {
      path: 'listalumnos/:id',
      title: 'Alumnos',
      loadComponent: () =>
        import('./administrator/institutions/liststudents-institutions/liststudents-institutions.component').then((c) => c.ListStudentsInstitutionsComponent),
    },
    {
      path: 'listamaestros/:id',
      title: 'Maestros',
      loadComponent: () =>
        import('./administrator/institutions/list-teachers-institutions/list-teachers-institutions.component').then((c) => c.ListTeachersInstitutionsComponent),
    },
  ],
];
```

Anexo 180. Routes para el dashboard de administrador (Parte 2)

```

ProyectoCEP > src > app > ui > app.routes.ts > routes > children > loadComponent
13   export const routes: Routes = [
146   children: [
206     {
207       path: 'Insteacher/:id',
208       title: 'Agregar maestro',
209       loadComponent: () =>
210         import('./administrator/teacherIns-register/teacherIns-register.component').then((c) => c.teacherInsRegisterComponent),
211     },
212     {
213       path: 'teacherGroups/:id',
214       title: 'Grupos',
215       loadComponent: () =>
216         import('./administrator/institutions/teacher-groups/teacher-groups.component').then((c) => c.TeacherGroupsComponent),
217     },
218     {
219       path: 'grupo/:grupoId/alumnos',
220       title: 'Lista del grupo',
221       loadComponent: () =>
222         import('./administrator/institutions/liststudent-group/liststudent-group.component').then((c) => ListstudentGroupComponent),
223     },
224     {
225       path: 'usuarios',
226       title: 'Administracion de usuarios',
227       loadComponent: () =>
228         import('./administrator/usuarios/adm-usuarios/adm-usuarios.component').then((c) => AdmUsuariosComponent),
229     },
230     {
231       path: 'crearpais',
232       title: 'Agregar Pais',
233       loadComponent: () =>
234         import('./administrator/Countries/add-countrie/add-countrie.component').then((c) => AddCountrieComponent),
235     },
236   ],

```

Anexo 181. Routes para el dashboard de administrador (Parte 3)

```

ProyectoCEP > src > app > ui > app.routes.ts > routes > children > loadComponent
13   export const routes: Routes = [
146   children: [
238     {
239       path: 'paises',
240       title: 'Paises',
241       loadComponent: () =>
242         import('./administrator/Countries/adm-countries/adm-countries.component').then((c) => AdmCountriesComponent),
243     },
244     {
245       path: 'instituciones/:countryId',
246       title: 'Instituciones',
247       loadComponent: () =>
248         import('./administrator/Countries/country-institutions/country-institutions.component').then((c) => CountryInstitutionsComponent),
249     },
250     {
251       path: 'quiz',
252       title: 'Cuestionarios',
253       loadComponent: () =>
254         import('./administrator/quiz/quiz-management/quiz-management.component').then((c) => QuizManagementComponent),
255     },
256     {
257       path: 'quiz',
258       title: 'Cuestionarios',
259       loadComponent: () =>
260         import('./administrator/quiz/quiz-management/quiz-management.component').then((c) => c.QuizManagementComponent),
261     },
262     {
263       path: 'create-quiz',
264       title: 'Crear cuestionario',
265       loadComponent: () =>
266         import('./administrator/quiz/create-quiz/create-quiz.component').then((c) => c.CreateQuizComponent),
267     },

```

Anexo 182. Routes para el dashboard de administrador (Parte 4)


```
    {
      path: 'quiz/:id/detalles',
      title: 'Detalles del cuestionario',
      loadComponent: () =>
        import('./administrator/quiz/quiz-details/quiz-details.component').then((c) => c.QuizDetailsComponent),
    },
  ],
},
```

Anexo 183. Routes para el dashboard de administrador (Parte 5)
