



TECNOLÓGICO
NACIONAL DE MÉXICO



INSTITUTO TECNOLÓGICO DE APIZACO

INFORME TÉCNICO DE RESIDENCIAS PROFESIONALES

“DESARROLLO DEL MÓDULO REGISTRO DEL DISTRIBUIDOR EN LA PLATAFORMA INTEGRAL PARA SEGUIMIENTO, CONTROL Y MEJORA DE PROCESOS DE PINTUMEX”

Julio Ernesto Montiel Pérez
No. De control: 19371032

ASESOR INTERNO:
Guadalupe Ortega Cruz

ASESOR EXTERNO
Cinthya Berruecos Méndez

Apizaco, Tlaxcala, **21/agosto /2024**

OFICIO No: SC/OPV/0812/24

ASUNTO: Liberación de proyecto
para titulación integral

CESAR REYNALDO RAMOS GÓMEZ
JEFE DEL DEPTO. DE DIVISIÓN DE ESTUDIOS PROFESIONALES
P R E S E N T E.

POR ESTE MEDIO LE INFORMO QUE HA SIDO LIBERADO EL SIGUIENTE PROYECTO PARA LA TITULACIÓN INTEGRAL

NOMBRE DEL EGRESADO:	JULIO ERNESTO MONTIEL PÉREZ
CARRERA:	INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIONES
NO. DE CONTROL:	19371032
NOMBRE DEL PROYECTO:	Desarrollo del módulo registro del distribuidor en la plataforma integral para seguimiento, control y mejora de procesos de Pintumex.
PRODUCTO U OPCIÓN:	TITULACIÓN INTEGRAL INFORME DE RESIDENCIA PROFESIONAL

AGRADEZCO DE ANTEMANO SU VALIOSO APOYO EN ESTA IMPORTANTE ACTIVIDAD PARA LA FORMACIÓN PROFESIONAL DE NUESTROS EGRESADOS.

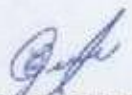
ATENTAMENTE

Excelencia en Educación Tecnológica
Pensar para Servir, Servir para Triunfar


JOSÉ ANTONIO CUATEPOTZO VARELA
JEFE DEL DEPTO. DE SISTEMAS Y COMPUTACIÓN

c.c.p. División de Estudios Profesionales
c.c.p. Depto. de Servicios Escolares
c.c.p. Archivo
JACV/mrol*


SECRETARÍA DE EDUCACIÓN PÚBLICA
TECNOLÓGICO NACIONAL
DE MÉXICO
INSTITUTO TECNOLÓGICO DE APIZACO
DEPARTAMENTO DE
SISTEMAS Y COMPUTACIÓN


GUADALUPE ORTEGA CRUZ
ASESOR INTERNO





Apizaco, Tlaxcala, **03/octubre/2024**

ASUNTO: AUTORIZACIÓN DE IMPRESIÓN.

MONTIEL PÉREZ JULIO ERNESTO
EGRESADO DE LA CARRERA DE INGENIERÍA
EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIONES.
CON NÚMERO DE CONTROL 19371032
PRESENTE.

Por este medio, me permito informar a usted que por aprobación de la comisión revisora asignada para valorar el trabajo que mediante la opción **TITULACIÓN INTEGRAL INFORME DE RESIDENCIA PROFESIONAL** cuyo tema es: **DESARROLLO DEL MÓDULO REGISTRO DEL DISTRIBUIDOR EN LA PLATAFORMA INTEGRAL PARA SEGUIMIENTO, CONTROL Y MEJORA DE PROCESOS DE PINTUMEX**, conforme a lo establecido en el procedimiento para la obtención del Título Profesional en el Instituto Tecnológico, la División de Estudios Profesionales a mi cargo le otorga:

AUTORIZACIÓN DE IMPRESIÓN

Debiendo entregar **2 USB** del mismo, apropiadamente encuadernados, a la brevedad, para presentar su acto de recepción profesional.

Sin otro particular por el momento me despido de Usted.

ATENTAMENTE

Excelencia en Educación Tecnológica
Pensar para Servir, Servir para Triunfar

CESAR REYNALDO RAMOS GÓMEZ
JEFE DE LA DIVISIÓN DE ESTUDIOS PROFESIONALES



SECRETARÍA DE EDUCACIÓN PÚBLICA
TECNOLÓGICO NACIONAL
DE MÉXICO
INSTITUTO TECNOLÓGICO DE APIZACO

**DIVISIÓN DE
ESTUDIOS PROFESIONALES**



Av. Instituto Tecnológico No. 418, San Andrés Ahuashuatepec, Municipio de Tzompantepec,
Tlaxcala, Mex. C.P. 90491 Tel. (241) 4172010 Ext. 142, e-mail: dep@apizaco.tecnm.mx



2024
Felipe Carrillo
PUERTO
GOBIERNO DEL ESTADO DE GUERRERO
SECRETARÍA DE EDUCACIÓN Y CULTURA
DIRECCIÓN GENERAL DE EDUCACIÓN

Agradecimientos

En primer lugar, les agradezco a mis padres que siempre me han brindado su apoyo incondicional para poder cumplir todos mis objetivos personales y académicos. Ellos son los que con su cariño me han impulsado siempre a perseguir mis metas y nunca abandonarlas frente a las adversidades. También son los que me han brindado el soporte material y económico para poder concentrarme en los estudios y nunca abandonarlos.

Les agradezco muy profundamente a mis tutores por su dedicación y paciencia, sin sus palabras y correcciones precisas no hubiese podido lograr llegar a esta instancia tan anhelada. Gracias por su guía y todos sus consejos, los llevaré grabados para siempre en la memoria en mi futuro profesional.

Finalmente, son muchos los docentes que han sido parte de mi camino universitario, y a todos ellos les quiero agradecer por transmitirme los conocimientos necesarios para hoy poder estar aquí. Sin ustedes los conceptos serían solo palabras, y las palabras ya sabemos quién se las lleva, el viento.

¡Muchas gracias!

Resumen

El presente proyecto aborda el desarrollo del módulo "Registro del Distribuidor" para la plataforma integral de Pintumex, destinada al seguimiento, control y mejora de sus procesos internos. El objetivo principal del módulo es facilitar la incorporación, gestión y seguimiento de los distribuidores, aspectos clave para la eficiencia operativa y la competitividad de la empresa.

El proyecto se desarrolla mediante metodologías ágiles, garantizando flexibilidad y respuesta rápida a los requisitos cambiantes. Se describe el proceso de diseño e implementación del módulo, incluyendo la selección de tecnologías, la definición de la arquitectura del sistema, y la integración con la plataforma existente. Además, se analizan los beneficios esperados, como la mejora en la toma de decisiones estratégicas y la optimización de la cadena de suministro.

Finalmente, se discuten los resultados obtenidos con la implementación del módulo, evidenciando cómo la digitalización de este segmento del proceso puede generar un impacto significativo en la eficiencia y transparencia de la gestión de distribuidores en Pintumex.

Índice

Agradecimientos.....	4
Resumen	5
Introducción	9
Generalidades de la empresa.....	10
Datos de la empresa SmartSoft America BA S.A de C.V.....	11
Problemas a Resolver	13
Objetivo General.....	13
Objetivos Específicos	13
Justificación	14
Marco Teórico	15
Metodología de desarrollo ágil SCRUM	15
Herramientas de trabajo	18
NetBeans	18
Navicat.....	19
TortoiseSVN	21
Minuta de trabajo	22
Java	23
PrimeFaces.....	24
Java server faces.....	26
GlassFish.....	26
Sistemas Operativos	30
Windows 10	30
Análisis de requerimientos.....	32
Requerimientos funcionales	33
Requerimientos no funcionales.....	37
Diccionario de datos	38
Diseño	43
Desarrollo	57
Resultados	72

Portal de distribuidores prospectos.....	72
Registro del distribuidor prospecto	73
Inicio de sesión.....	74
Bienvenida al sistema de administración	75
Proceso de clientes	75
validación de datos.....	77
Pruebas unitarias.....	78
Conclusiones.....	81
Experiencia laboral	83
Competencias desarrolladas	83
Referencias consultadas	85

Índice de figuras y tablas

Figura 1 Ubicación Geográfica de la empresa	10
Figura 2 Organigrama de la empresa	11
Figura 3 Flujo de Scrum en un Sprint	16
Figura 4 Características de navicat	20
Figura 5 Características de Glassfish	27
Figura 6 Minuta de trabajo	41
Figura 7 Diagrama de CU – Gestión de distribuidor prospecto	43
Figura 8 Diagrama de CU – Gestión del administrador.....	44
Figura 9 Diagrama de Secuencia – Gestión de distribuidores prospectos	46
Figura 10 Diagrama de Secuencia – Gestión de administrador	48
Figura 11 Módulo portal de registro	51
Figura 12 Formulario de registro.....	52
Figura 13 Inicio de sesión.....	53
Figura 14 Bienvenida del sistema de administración	54
Figura 15 Prospectos clientes.....	55
Figura 16 Validación de datos	56
Figura 17 Carga de los Módulos de pintumex.....	57
Figura 18 Implementación de la base de datos	58
Figura 19 Demostración de subida de cambios usando SVN	59
Figura 20 Código XHTML del portal	60
Figura 21 Clase entity.....	61
Figura 22 Clase View	62
Figura 23 Clase controller.....	63
Figura 24 Clase business	66

Figura 25 Clase DAO	70
Figura 26 Módulo de portal de registro	73
Figura 27 Formulario de registro.....	74
Figura 28 Módulo de inicio de sesión del administrador	74
Figura 29 Bienvenida al sistema de pintumex.....	75
Figura 30 Pantalla de proceso de clientes	76
Figura 31 Validación de datos	77
Figura 32 Prueba de validación del comportamiento del método findByCodigoPostal ...	78
Figura 33 Prueba de validación de campos	79
Figura 34 Prueba de guardado de ubicación	79
 Tabla 1 Ventajas y desventajas de scrum	 17
Tabla 2 Características de NetBeans	18
Tabla 3 Ventajas y desventajas de NetBeans.....	19
Tabla 4 Ventajas y desventajas de Navicat	21
Tabla 5 Características, ventajas y desventajas TortoiseSVN	22
Tabla 6 Características, ventajas y desventajas de java.....	24
Tabla 7 Características, ventajas y desventajas de primefaces	25
Tabla 8 Ventajas y desventajas de Glassfish.....	28
Tabla 9 Características, ventajas y desventajas diagramas de secuencia.....	29
Tabla 10 Características, ventajas y desventajas de diagramas de caso de uso	30
Tabla 11 Características, ventajas y desventajas de Windows 10	31
Tabla 12 Requerimientos funcionales del registro del distribuidor prospecto	35
Tabla 13 Requerimientos funcionales de gestión del administrador	36
Tabla 14. Requerimientos no funcionales	37
Tabla 15 Diccionario de datos	38
Tabla 16 Asignación de actividades	40

Introducción

En la era digital actual, la eficiencia y efectividad de los procesos empresariales dependen en gran medida de la capacidad de las organizaciones para integrar tecnologías que faciliten y mejoren la gestión y supervisión de sus operaciones. Pintumex, una compañía líder en la industria de la pintura enfrenta el reto constante de optimizar sus procesos internos para sostener su competitividad en el mercado. Una de las áreas críticas que requiere atención es la gestión de sus distribuidores, quienes juegan un papel crucial en la cadena de suministro y la satisfacción del cliente final.

Este trabajo se enfoca en el desarrollo del módulo "Registro del Distribuidor" dentro de la plataforma integral de Pintumex, diseñada para el seguimiento, control y mejora de procesos. La creación de este módulo es esencial no solo para simplificar y estandarizar la entrada de nuevos distribuidores al sistema, sino también para asegurar un seguimiento efectivo y una gestión detallada de los datos de los distribuidores existentes. La integración de este módulo promete facilitar la toma de decisiones estratégicas, mejorar la eficiencia operativa y aumentar la transparencia en las operaciones de distribución.

Este proyecto detallará el diseño, desarrollo e implementación del módulo, utilizando metodologías de desarrollo de software ágiles para asegurar su adaptabilidad y eficacia. Se explorará cómo la incorporación de tecnologías avanzadas y prácticas de ingeniería de software puede transformar positivamente la gestión de relaciones con distribuidores en Pintumex, estableciendo un precedente significativo en la modernización de procesos en la industria de la pintura.

1. Generalidades de la empresa

Empresa: **SmartSoft America BA**

Dirección: C. Adolfo López Mateos 131A, T

Colonia: Texcacoac, 90806 Chiautempan, Tlax. Tel: 246 117 5228

Correo electrónico: rbarrientosmartsoft@gmail.com

Página web: **smartsoftamerica.com.mx**

Ubicación Geográfica de la empresa

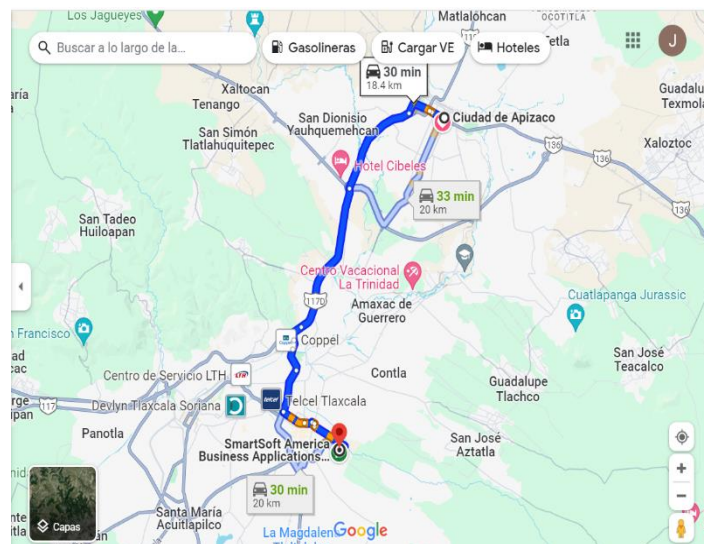


Figura 1 Ubicación Geográfica de la empresa

Organigrama de la empresa

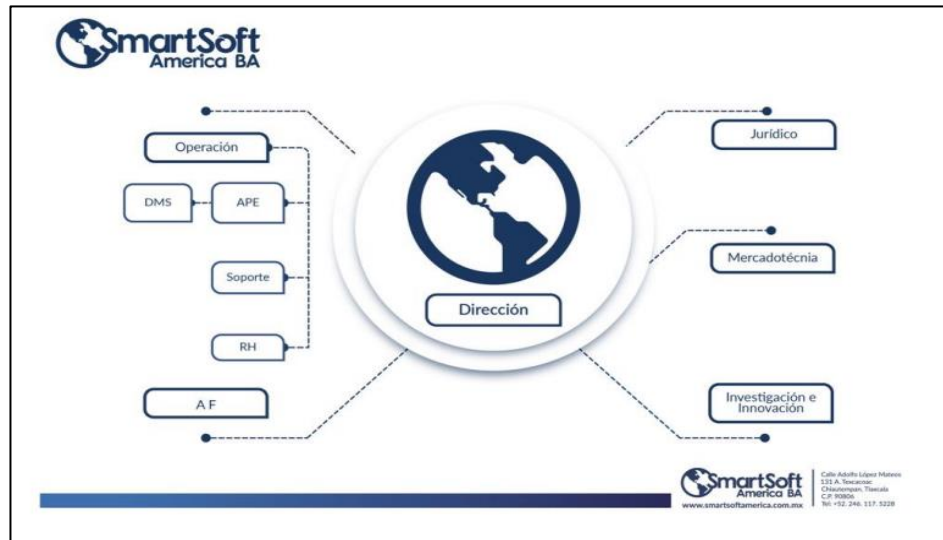


Figura 2 Organigrama de la empresa

Datos de la empresa SmartSoft America BA S.A de C.V

En SmartSoft America BA, se aplica el motor del cambio desde la fundación de la empresa en mayo de 2009, crea productos con altos estándares tecnológicos, actualmente en el mercado cuenta con productos que dan solución a las necesidades de sus clientes con sistemas enfocados a la Industria 4.0, de Internet de las Cosas, para PyMES, para el sector gubernamental, también para logística y diferentes Apps disponibles en Android y en IOS. Destacando el DESARROLLO DE SOFTWARE a la medida, como un servicio con calidad internacional.

Misión

Es una empresa de Investigación, Innovación, y Desarrollo Tecnológico enfocado a crear y mejorar productos y servicios de Software e Industria 4.0 de calidad que permite incrementar el desempeño, competitividad y satisfacción de sus clientes.

Visión

Ser una organización de clase mundial que brinde productos y servicios de Software e Industria 4.0 cumpliendo con los más altos estándares de calidad, compromiso y eficiencia en beneficio de sus clientes, tomando como base la investigación, innovación y el desarrollo tecnológico.

Valores

Los valores son principios que permiten orientar el comportamiento en función de realizarse como personas:

- Compromiso
- Lealtad
- Honestidad y respeto
- Comunicación
- Profesionalismo
- Trabajo en equipo
- Libertad de innovación

Reconocimientos y certificaciones

- Evaluados CMMI DEV/5 y CMMI SVC/5
- Norma de Calidad para Desarrollo de Software MOPROSOFT (NMX-I-059/01-NYCE-2005).
- Cuenta con el Registro Nacional de Instituciones y Empresas Científicas y Tecnológicas (RENIECYT) por parte de CONACyT.
- Empresa afiliada al Clúster de Tecnologías de la Información del estado de Tlaxcala.

2. Problemas a Resolver

Automatizar el proceso de adquisición y distribución de datos para la generación de informes que actualmente se llevan a cabo de manera manual.

El módulo de registros de distribuidores prospectos dentro de la plataforma integral de Pintumex tiene el propósito de llevar un registro preciso y sistemático de los clientes que desean ser distribuidores asociados, facilitando así su gestión y seguimiento.

3. Objetivo General

- Desarrollar un módulo de registro de distribuidores dentro de la plataforma integral de Pintumex, con el propósito de llevar un registro preciso y sistemático de los clientes que desean ser distribuidores asociados, facilitando así su gestión y seguimiento.

Objetivos Específicos

- Simplificar el proceso de registro para los distribuidores prospectos, permitiéndoles ingresar fácilmente la información requerida en la plataforma de Pintumex.
- Mantener y mejorar la comunicación entre Pintumex y sus distribuidores prospectos en donde para el cliente sea más interactiva la gestión de información y consultas relacionadas con el registro.
- Incluir la participación de los distribuidores prospectos y mejorar el proceso de la adquisición y distribución de datos para la generación de informes, permitiendo que Pintumex evalúe el desempeño de su red de distribución.

- Mejorar los procesos de Pintumex relacionados con el registro de distribuidores prospectos, permitiendo una gestión más eficiente de los recursos y reduciendo los tiempos de respuesta.

4. Justificación

Se pretende aplicar las nuevas tecnologías con el módulo de distribuidores prospectos logrando mejorar la gestión de distribuidores, optimizar procesos internos y ofrecer un servicio más personalizado. Esto fortaleciendo la competitividad de la empresa y satisfacción del cliente.

5. Marco Teórico

Para alcanzar con éxito el desarrollo de este proyecto, se utilizará la metodología ágil *Scrum* como marco de trabajo principal. Esta metodología se caracteriza por su enfoque iterativo e incremental, lo que permite una adaptación fluida a los cambios en los requisitos del proyecto a medida que este avanza.

En cuanto a las herramientas seleccionadas para este proyecto, se ha optado por *Java 8.2* como el lenguaje primario de programación, reconocido por su versatilidad y robustez en el desarrollo de aplicaciones empresariales. Para la interfaz de usuario, se están utilizando *PrimeFaces* y *JavaServer Faces (JSF)*, componentes que ofrecen una amplia gama de opciones para la creación de interfaces interactivas y atractivas. Además, se empleó *Figma* para crear prototipos, interfaces de sitios y aplicaciones, presentaciones, logotipos y realizar pruebas de usabilidad.

Por otro lado, *GlassFish* se ha elegido como servidor de aplicaciones, proporcionando un entorno de ejecución confiable y escalable para desplegar la aplicación desarrollada. Esta combinación de metodología y herramientas garantiza no solo una entrega eficiente y oportuna del producto, sino también la construcción de una aplicación web sólida y adaptable, capaz de satisfacer las necesidades cambiantes del cliente y los usuarios finales.

5.1 Metodología de desarrollo ágil SCRUM

Qué es SCRUM

SCRUM tiene el significado de colaboración, ya que es un término sacado del rugby donde los jugadores se apoyan mutuamente. La metodología *agile SCRUM* es un entorno de trabajo donde se aplican una serie de acciones con el objetivo de abordar proyectos de forma mucho más rápida y eficiente.

En SCRUM el trabajo colaborativo es fundamental, por lo que se debe garantizar un alto nivel de comunicación entre los distintos componentes que participan en el proyecto. SCRUM trabaja con Sprints, es decir, hay que seguir una serie de fases para abordar cada tarea, una fortaleza clave de Scrum es el uso de equipos multidisciplinarios, autoorganizados y empoderados que dividen su trabajo en ciclos de trabajo cortos y concentrados llamados *sprints*. La **figura 3** proporciona una visión general de flujo de un proyecto Scrum.

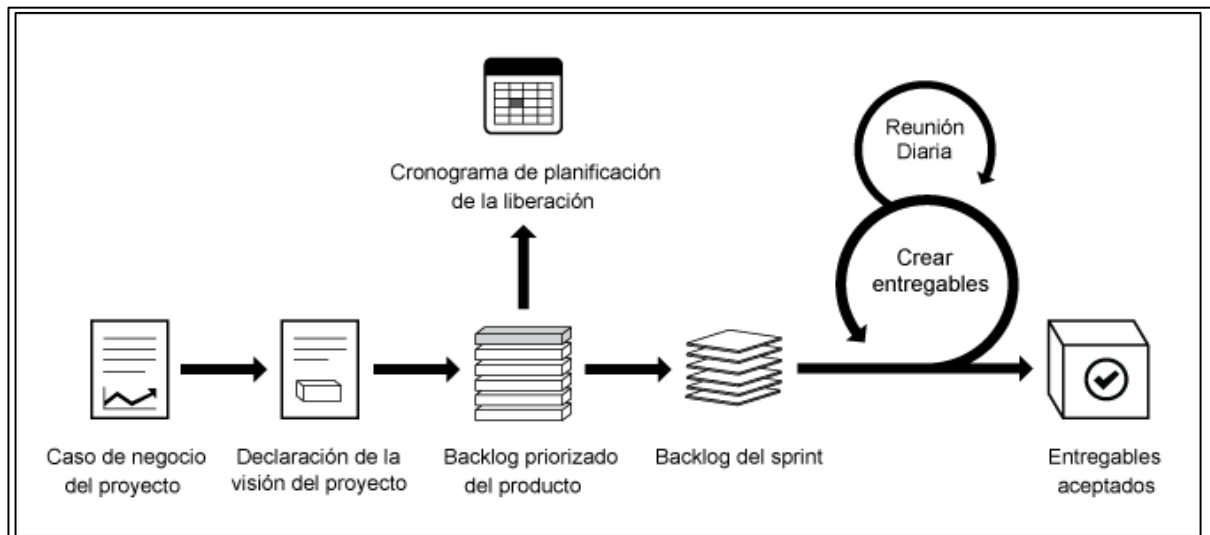


Figura 3 Flujo de Scrum en un Sprint

El ciclo de Scrum empieza con una reunión de interesados del negocio, durante la cual se crea la visión del proyecto. Después, el Product Owner desarrolla un backlog priorizado del producto que contiene una lista requerimientos del negocio y del proyecto por orden de importancia en forma de una historia de usuario.

Cada sprint inicia con una reunión de planificación del sprint, donde se ponen a consideración las historias de usuario de mayor prioridad para ser incluidas en el sprint. Un sprint generalmente tiene una duración de una a cuatro semanas en las cuales el Equipo Scrum trabaja en la creación de entregables potencialmente enviabiles o incrementos del producto. Durante el sprint, se llevan cabo Daily Standups muy breves y concretos, donde los miembros del equipo discuten el avance diario. Hacia el final del sprint, se lleva a cabo una reunión de revisión del sprint en la cual se hace una demostración de los entregables al Product Owner y a los interesados del negocio relevantes. El Product Owner acepta los entregables solo si cumplen con los criterios de aceptación predefinidos. El ciclo del sprint concluye con una reunión de retrospectiva del sprint, donde el equipo analiza la forma de mejorar su trabajo y su desempeño a medida que avanza al siguiente sprint, la siguiente **tabla 1** muestra las ventajas y desventajas de scrum. (Pre, 18)

Tabla 1 Ventajas y desventajas de scrum

Ventajas	Desventajas
Permite crear productos listos para salir al mercado en un tiempo menor.	Si los objetivos no están definidos correctamente, habrá ambigüedad en todo el proceso.
Ofrece el marco necesario para cambiar de rumbo o de enfoque cuando un proyecto así lo necesita.	Si el cliente tiene demandas nuevas en partes avanzadas del proceso, puede estancarse todo el proyecto, por lo que es necesario establecer fechas límite.
Ayuda a comunicarse de manera interna al mantener reuniones diarias que ayudan a retroalimentar el proceso y encontrar los errores.	Si no hay confianza en el equipo, los controles excesivos disminuirán el rendimiento y desvirtuarán el propósito de este marco de trabajo.
Permite evaluar el rendimiento individual y del equipo, así como para proporcionar su mejora.	Debido a que es una forma de trabajo exigente, es necesario contar con miembros de gran experiencia o capacidad; de lo contrario, el proyecto puede fallar.
Fomenta la revisión por parte del cliente y sus necesidades pueden ser gestionadas de manera rápida.	Se aplica a equipos reducidos

Herramientas de trabajo

5.2.1 NetBeans

NetBeans es un IDE o entorno de desarrollo integrado, basado en el lenguaje Java y ejecutado en Swing. De esta forma, NetBeans o Apache NetBeans es una aplicación de código abierto, que ha cobrado bastante popularidad en los últimos años.

Este IDE, orientado principalmente a las apps de Java, ofrece diferentes herramientas digitales como editor de texto, código, compilador, interfaz gráfica de usuario; además de un depurador.

Por otro lado, cabe destacar que NetBeans facilita la creación de aplicaciones estructuradas, ya que están basadas en un conjunto de módulos. Así, se favorece el desarrollo de las diversas funciones de una manera independiente y pudiendo también reutilizar los componentes.

NetBeans ayuda en el desarrollo de cualquier tipo de software y en sus distintas fases. A los programadores les sirve como medio para la escritura, compilación o, simplemente, para hallar errores en programas basados en los anteriores lenguajes. La siguiente **Tabla 2** presenta las Características de NetBeans, así como la **tabla 3**. Ventajas y desventajas). (Oracle, 2023)

Tabla 2 Características de NetBeans

Características	Descripción
Editor de código	Ofrece un editor de código con resaltado de sintaxis, completado automático y refactorización.
Depuración	Proporciona herramientas de depuración para encontrar y corregir errores en el código.
Administrador de proyectos	Permite gestionar y organizar proyectos de desarrollo de software de manera eficiente.
Soporte para múltiples lenguajes	Además de Java, ofrece soporte para otros lenguajes como HTML, CSS, JavaScript, PHP, entre otros
Integración con servidores de aplicaciones	Permite el despliegue y la administración de aplicaciones en servidores como Apache Tomcat y GlassFish.
Integración con Git	Permite la integración con el sistema de control de versiones Git para el seguimiento de cambios.

Tabla 3 Ventajas y desventajas de NetBeans

Ventajas	Desventajas
Entorno de desarrollo integrado (IDE) completo	Puede consumir más recursos de sistema en comparación con IDEs más ligeros
Amplia gama de características y herramientas	Interfaz de usuario puede resultar abrumadora para algunos usuarios nuevos
Fuerte comunidad y soporte	Algunas características pueden ser menos intuitivas de usar
Facilidad de uso para proyectos Java	Puede presentar problemas de rendimiento con proyectos muy grandes
Amplia compatibilidad con frameworks y tecnologías	Actualizaciones y mejoras pueden ser menos frecuentes que en otras IDEs
Posibilidad de desarrollar aplicaciones de escritorio, web y móviles	Requiere instalación y configuración inicial
Gran cantidad de plugins disponibles	Puede requerir más recursos de memoria y espacio en disco
Herramientas de depuración avanzadas	Algunos plugins pueden ser menos estables o tener menos soporte
Actualizaciones regulares y correcciones de errores	La documentación puede ser menos completa o estar desactualizada en algunos casos

5.2.2 Navicat

Es un Administrador de Base de datos rápido, fiable y asequible. Especialmente diseñada para simplificar la gestión de bases de datos y reducir los costes de administración. Con una interfaz gráfica del usuario intuitiva, *Navicat* proporciona una manera más fácil de gestionar, diseñar y manipular datos en MySQL, MariaDB, SQL Server, SQLite, Oracle y PostgreSQL.

Contiene una Interfaz de Usuario Intuitiva y bien diseñada para crear, modificar y gestionar todos los objetos de la base de datos, como tablas, vistas, funciones / procedimientos, índices, triggers (disparadores) y secuencias.

El diseñador de tablas ayuda a crear y modificar tablas de la base de datos permitiendo diseñar y visualizar una sola tabla en la base de datos a la que esté conectado, ayuda a configurar opciones avanzadas como las relaciones, restricciones, disparadores y más, las ventajas y desventajas de utilizar Navicat se pueden visualizar claramente en la **Tabla 4**, donde se visualizan los beneficios y las posibles limitaciones de la herramienta. Además, las características principales de Navicat se resumen de manera simplificada en la **Figura 4**, facilitando una comprensión rápida de sus capacidades y funcionalidades. (Partner, 2024)

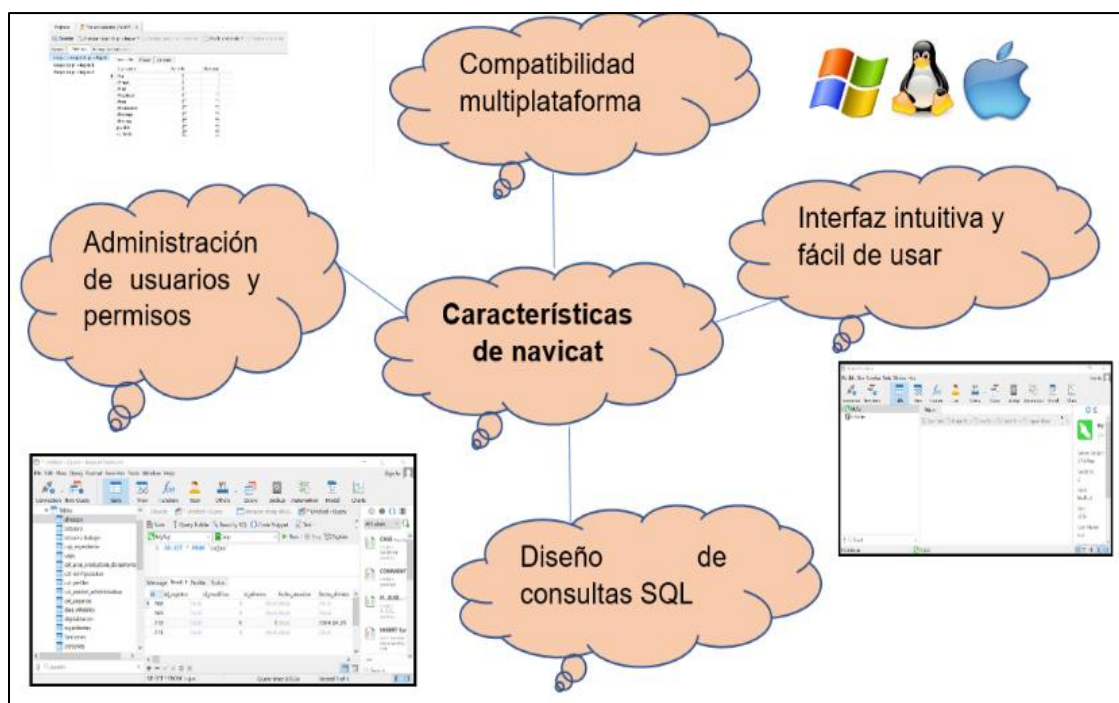


Figura 4 Características de navicat

Tabla 5 Ventajas y desventajas de Navicat

Ventajas	Desventajas
Soporte para múltiples bases de datos.	Licencia de pago, lo que puede ser costoso.
Herramientas avanzadas de administración de bases de datos.	Puede ser abrumador para usuarios principiantes.
Funcionalidades de importación y exportación de datos.	Requiere cierta curva de aprendizaje.
Soporte técnico y actualizaciones regulares.	Requiere recursos de sistema significativos.
Generación de informes y análisis de datos.	Puede experimentar problemas de rendimiento con grandes conjuntos de datos.
Seguridad avanzada con funciones de encriptación y autenticación.	Algunas características pueden no estar disponibles en todas las plataformas.
Funcionalidades de copia de seguridad y restauración.	Soporte limitado para bases de datos NoSQL
Editor SQL avanzado con resaltado de sintaxis	Actualizaciones ocasionales problemáticas

5.2.3 TortoiseSVN

Es un cliente gratuito de código abierto para el sistema de control de versiones Apache Subversion. Esto significa que TortoiseSVN administra archivos y directorios a lo largo del tiempo. Los archivos se almacenan en un repositorio central, tortoiseSVN está desarrollado con la intención de ser una extensión del Explorador de archivos de Windows. Por este motivo, puede que en otras aplicaciones la integración no sea tan completa, y que, por ejemplo, los iconos superpuestos no se muestren.

El estado de cada carpeta y archivo versionado se indica por pequeños iconos superpuestos. De esta forma, puede ver fácilmente el estado en el que se encuentra su copia de trabajo.

Todos los comandos de Subversión están disponibles desde el menú contextual del explorador. TortoiseSVN añade su propio submenú allí, las ventajas y desventajas de utilizar TortoiseSVN se pueden visualizar claramente en la **Tabla 5**, donde se muestran los beneficios y posibles limitaciones de esta herramienta, así como sus características y descripción. (Perez F. , 2023)

Tabla 5 Características, ventajas y desventajas TortoiseSVN

Características	Descripción	Ventajas	Desventajas
Cliente de control de versiones SVN	TortoiseSVN es un cliente de control de versiones basado en SVN (Subversión), que permite gestionar	Integración con el explorador de archivos	Interfaz gráfica puede ser menos intuitiva para usuarios no familiarizados con SVN
Integración con el explorador de archivos	Ofrece integración con el explorador de archivos de Windows, lo que permite realizar operaciones de	Acceso rápido a operaciones de control de versiones desde el explorador	Limitado a entornos Windows
Funciones básicas de control de versiones	Permite realizar operaciones básicas como actualizar, revertir cambios, confirmar y fusionar cambios.	Gestión eficiente de versiones de archivos y carpetas	Limitado en funcionalidades avanzadas comparado con clientes de control de versiones más complejos
Soporte para múltiples idiomas	TortoiseSVN está disponible en varios idiomas, que facilita su uso para usuarios de diferentes	Amplia comunidad de usuarios y recursos disponibles en línea	Algunas traducciones pueden no estar completamente actualizadas o ser menos precisas

5.2.4 Minuta de trabajo

Una minuta de trabajo es un documento que resume de manera concisa los puntos tratados, decisiones tomadas y acciones acordadas durante una reunión. Este documento es esencial para asegurar que todos los participantes y otras partes interesadas estén informados sobre lo discutido y puedan hacer un seguimiento de las tareas asignadas.

La minuta sirve como un registro formal que facilita la comunicación y la transparencia dentro de una organización, así como la continuidad de los proyectos y la rendición de cuentas.

La estructura de una minuta de trabajo generalmente incluye varios elementos clave. En primer lugar, se mencionan los datos básicos de la reunión, como la fecha, hora, lugar y lista de asistentes. Estos detalles iniciales son importantes para contextualizar la reunión y registrar la participación de los miembros. También puede incluir el nombre del moderador o líder de la reunión y del redactor de la minuta.

Se presenta el orden del día o agenda de la reunión. Este apartado lista los temas que se iban a tratar, proporcionando un marco de referencia para entender el desarrollo de la sesión. Cada tema del orden del día se aborda en la sección siguiente, donde se describen los puntos principales discutidos. Aquí, es fundamental ser claro y preciso, capturando la esencia de las discusiones sin entrar en detalles excesivos.

Un componente crucial de la minuta es el registro de las decisiones tomadas. Este segmento debe identificar claramente qué decisiones se acordaron y, cuando sea pertinente, quiénes fueron los responsables de dichas decisiones.

Esto no solo ayuda a recordar lo que se decidió, sino que también permite que se asigne responsabilidad y se realice un seguimiento efectivo de las tareas y compromisos. (Juarez, 2023)

5.2.5 Java

Java es un lenguaje sencillo y orientado a objetos, que permite el desarrollo de aplicaciones en diversas áreas, como seguridad, animación, acceso a bases de datos, aplicaciones cliente-servidor, interfaces gráficas, páginas Web interactivas y desarrollo de aplicaciones móviles, entre otras. Una de sus principales características es la creación de módulos reutilizables de una manera ágil, que funcionan sin la necesidad de conocer su estructura interna. Esto permite al usuario añadir nuevos módulos, y obtener programas independientes de la plataforma en la cual fueron desarrollados, gracias a la Máquina Virtual con la que cuenta Java. Para más información visualice en la **Tabla 6**. De características ventajas y desventajas). (IBM, ¿Que es Java?, 2023)

Tabla 6 Características, ventajas y desventajas de java

Características	Descripción	Ventajas	Desventajas
Portabilidad	Permite ejecutar el mismo código en múltiples plataformas sin necesidad de recompilar.	Facilita el desarrollo de aplicaciones que pueden ser utilizadas en diferentes sistemas operativos.	La ejecución a través de la Máquina Virtual Java (JVM) puede introducir cierta sobrecarga de rendimiento.
Orientado a objetos	Java es un lenguaje completamente orientado a objetos, lo que significa que todo en Java es un objeto.	Fomenta la reutilización de código y el modularidad.	Puede requerir una curva de aprendizaje para comprender los conceptos de la programación orientada a objetos.
Seguridad	Java incorpora un robusto modelo de seguridad que protege contra amenazas como el código malicioso.	Proporciona un entorno seguro para ejecutar aplicaciones.	La estricta aplicación del modelo de seguridad puede limitar algunas funcionalidades y requerir ajustes adicionales.
Gestión automática de memoria	Java utiliza un sistema de recolección de basura (garbage collector) que simplifica la administración de la memoria y previene las fugas de memoria.	Evita las fugas de memoria y simplifica la administración de la memoria.	a recolección de basura puede introducir pausas en la ejecución del programa, lo que puede afectar el rendimiento en aplicaciones de tiempo real.
Desarrollo web	Java proporciona tecnologías como JavaServer Pages (JSP), Servlets y frameworks como Spring para el desarrollo de aplicaciones web escalables y dinámicas.	Permite el desarrollo de aplicaciones web completas con funcionalidades avanzadas.	La curva de aprendizaje puede ser empinada para los desarrolladores nuevos en el desarrollo web en Java, especialmente al trabajar con tecnologías avanzadas y frameworks.

5.2.6 PrimeFaces

PrimeFaces es una biblioteca de componentes para JavaServer Faces (JSF) de código abierto que cuenta con un conjunto de componentes enriquecidos que facilitan la creación de las aplicaciones web. Algunos de los componentes son los siguientes, editor de HTML, autocompletar, cartas, gráficas o paneles, entre otros, también cuenta con un Soporte de Ajax con despliegue parcial, lo que permite controlar qué componentes de la página actual se actualizarán y cuáles no, así como con 25 temas prediseñados y componentes para desarrollar aplicaciones web para teléfonos móviles, especiales para iPhones, Palm, Android y teléfonos móviles Nokia. (PrimeFaces, 2024) para más información visualice en la **Tabla 7**. De características, ventajas y desventajas).

Tabla 7 Características, ventajas y desventajas de primefaces

Características	Descripción	Ventajas	Desventajas
Componentes UI	Framework que ofrece una amplia gama de componentes de interfaz de usuario (UI) predefinidos y ricos para el desarrollo de aplicaciones web.	Facilita la construcción de interfaces de usuario modernas y atractivas.	La amplia variedad de componentes puede resultar abrumadora para los desarrolladores nuevos.
Ajax	Integración completa con Ajax para realizar actualizaciones de partes específicas de la página web de forma dinámica, sin necesidad de recargar la página completa.	Mejora la experiencia del usuario al proporcionar respuestas rápidas y sin interrupciones.	Requiere un buen entendimiento de Ajax para aprovechar al máximo esta característica.
Temas	Soporta temas personalizables que permiten adaptar la apariencia de los componentes a las necesidades del proyecto o a las especificaciones de diseño.	Permite la creación de interfaces de usuario con estilos visualmente atractivos.	La personalización puede ser limitada en comparación con otras soluciones más flexibles.
Facilidad de uso	Diseñado con una sintaxis simple y clara para la definición de componentes, lo que facilita su uso.	Facilita el desarrollo web al simplificar la creación y configuración de componentes.	Algunas características avanzadas pueden requerir una comprensión más profunda del framework.
Extensibilidad	Permite la creación de componentes personalizados que pueden integrarse con PrimeFaces, lo que ofrece flexibilidad para adaptar el framework.	Proporciona la capacidad de ampliar y personalizar.	Crear componentes personalizados puede requerir un tiempo y esfuerzo considerable.
Documentación	Amplia documentación disponible que incluye guías, tutoriales y ejemplos.	Facilita el aprendizaje y la comprensión del framework para desarrolladores de todos los niveles de experiencia.	La documentación puede no cubrir todas las situaciones y escenarios específicos del proyecto.

5.2.7 Java server faces

La tecnología JavaServer Faces es un framework de interfaz de componentes de usuarios del lado del servidor para las aplicaciones web basadas en la tecnología Java.

Los principales componentes de la tecnología JSF son:

- Una API para: Representar componentes de Interfaz de Usuario (UI) y gestionar su estado.
 - Manejar eventos, validar en el servidor y conversión de datos.
 - Definir la navegación de páginas.
 - Soporte de internacionalización y accesibilidad.
- Dos librerías de etiquetas JSP personalizadas para expresar componentes en una página JSP y enlazar los componentes a objetos del servidor.

JavaServer Faces tiene una infraestructura de interfaz de usuario o una interfaz de programación de aplicaciones (API) que facilita el desarrollo de aplicaciones web Java. Facilita la construcción de una interfaz de usuario a partir de un conjunto de componentes de interfaz de usuario reutilizables, así como también simplifica la migración de datos de aplicación hacia la interfaz de usuario o desde ésta. (IBM, ¿Que es Java?, 2023)

5.2.8 GlassFish

GlassFish es un servidor de aplicaciones de código abierto desarrollado por Oracle Corporation que implementa las especificaciones de la Plataforma de Servidores de Aplicaciones Java Enterprise Edition (Java EE). Está diseñado para ser utilizado en el desarrollo, implementación y ejecución de aplicaciones empresariales Java EE.

GlassFish un servidor para poder desarrollar tanto aplicaciones Java Platform Enterprise Edition (Java EE) como webs Java Web Services.

Fue creado en base a GlassFish Server Open Source Edition, un servidor de aplicaciones de código abierto de la comunidad de GlassFish. Oracle GlassFish Server ofrece un servidor de aplicaciones Java EE flexible, ligero y listo para la producción.

Es ampliamente utilizado en la industria para desarrollar y desplegar aplicaciones empresariales Java EE. Es especialmente popular entre los desarrolladores que prefieren soluciones de código abierto y buscan una alternativa gratuita a otros servidores de aplicaciones Java EE comerciales. (Services, 2021), para más información visualice en la **Tabla 8.** de ventajas y desventajas y en la **figura 5** las características de este mismo. (Services, 2021)

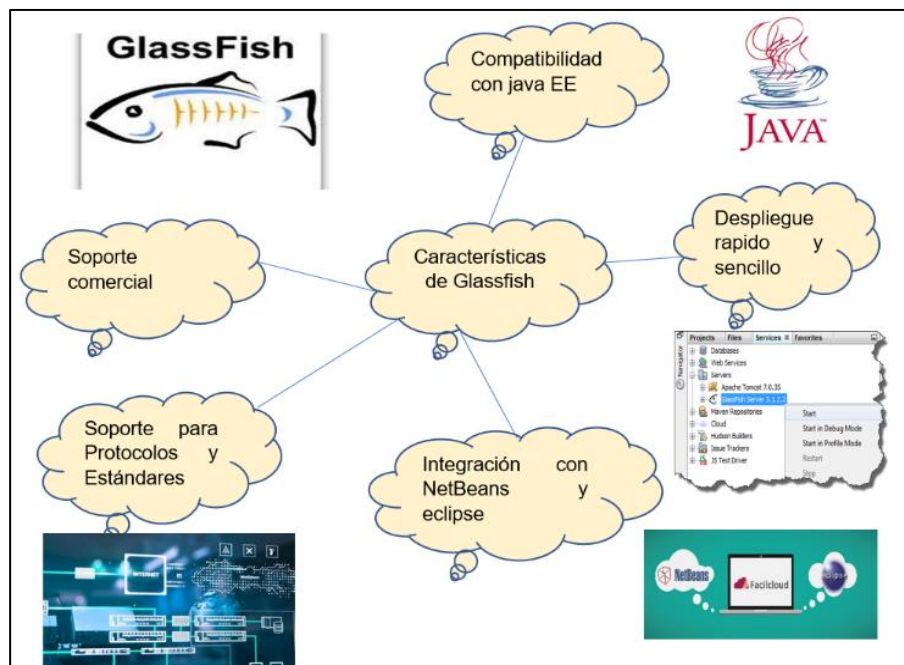


Figura 5 Características de Glassfish

Tabla 8 Ventajas y desventajas de Glassfish

ventajas	Desventajas
Implementación completa de Java EE	Requiere recursos de memoria y CPU significativos, que puede afectar el rendimiento en entornos con recursos limitados.
Código abierto	La curva de aprendizaje puede es empinada para los principiantes en cuanto a las configuraciones y opciones disponibles.
Escalabilidad horizontal y vertical	La documentación es insuficiente o desactualizada, lo que dificulta la resolución de problemas y la implementación de configuraciones avanzadas.
Herramientas de administración y monitoreo	La integración con IDEs es no tan fluida como con otros servidores de aplicaciones, lo que afecta la productividad del desarrollador.

5.2.9 Diagrama de secuencia

Un diagrama de secuencia UML es una representación gráfica que muestra la interacción de objetos en un sistema a lo largo del tiempo. Estos diagramas capturan la secuencia de mensajes intercambiados entre objetos y el orden en que ocurren estas interacciones, presentándolos como líneas de vida verticales y flechas horizontales. Esencialmente, los diagramas de secuencia ofrecen una visualización cronológica de las interacciones de objetos, brindando perspectivas sobre el comportamiento dinámico y flujo de un sistema, muestran la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada caso de uso.

Típicamente se examina la descripción de un caso de uso para determinar qué objetos son necesarios para la implementación del escenario. (visualícese en la **Tabla 9**. De características ventajas y desventajas). (miro, 2024)

Tabla 9 Características, ventajas y desventajas diagramas de secuencia

Características	Descripción	Ventajas	Desventajas
Representación visual de interacciones	Los diagramas de secuencia representan la interacción entre objetos en un sistema a través del tiempo, mostrando el flujo de mensajes entre ellos.	Claridad en la comprensión de la interacción entre objetos. Permite identificar fácilmente el orden de las operaciones.	Puede resultar complejo para sistemas con interacciones muy detalladas.
Modelado de comportamiento	Estos diagramas permiten modelar el comportamiento dinámico de un sistema, mostrando cómo los objetos interactúan entre sí en diferentes escenarios.	Ayuda a identificar y comprender el comportamiento del sistema en diferentes situaciones.	Requiere un buen entendimiento del sistema para crear diagramas precisos.
Soporte en herramientas de modelado	Hay una variedad de herramientas de modelado que admiten la creación de diagramas de secuencia, lo que facilita su creación y mantenimiento.	Ofrece una amplia gama de opciones para la creación y edición de diagramas.	Las funcionalidades avanzadas pueden requerir suscripciones o versiones premium de las herramientas.

5.2.10 Diagrama de casos de uso

Los diagramas de caso de uso son una herramienta utilizada en el análisis y diseño de sistemas informáticos y de software para representar las interacciones entre los usuarios (llamados actores) y el sistema en cuestión. Estos diagramas se centran en las acciones o funciones que un sistema debe realizar para satisfacer las necesidades de sus usuarios, sin adentrarse en los detalles de implementación técnica. En esencia, los diagramas de caso de uso proporcionan una visión clara y comprensible de cómo los usuarios interactúan con el sistema y qué funcionalidades se requieren para satisfacer sus necesidades.

En lugar de ello, se centran en representar las interacciones entre los usuarios y el sistema en términos de escenarios o casos de uso específicos. (FANDOM, 2023) (visualícese en la **Tabla 10**. De características ventajas y desventajas).

Tabla 10 Características, ventajas y desventajas de diagramas de caso de uso

Características	Descripción	Ventajas	Desventajas
Comunicación clara	Son una herramienta efectiva para comunicar los requisitos del sistema entre los diferentes stakeholders del proyecto, como desarrolladores, diseñadores y clientes.	Facilitan la comprensión compartida de los objetivos y funcionalidades del sistema.	Requieren una correcta interpretación y definición de los casos de uso para evitar malentendidos
Enfoque en funcionalidades	Los diagramas de casos de uso se centran en las funciones o acciones que el sistema debe realizar para satisfacer las necesidades de los usuarios	Mantienen el enfoque en los requisitos del usuario en lugar de en detalles técnicos.	Pueden ser limitados para representar aspectos no funcionales del sistema, como rendimiento o seguridad.
Herramienta de diseño	Son esenciales durante el proceso de análisis y diseño de sistemas, ayudando a modelar y comprender las interacciones entre usuarios y sistema.	Facilitan la visualización y comprensión de las interacciones entre usuarios y sistema.	Requieren experiencia en el proceso de análisis y diseño para crear diagramas efectivos.

6. Sistemas Operativos

6.1 Windows 10

Windows 10 es la última iteración del sistema operativo emblemático desarrollado por Microsoft, que ofrece una plataforma robusta y versátil para una amplia gama de dispositivos, desde computadoras de escritorio y portátiles hasta tabletas, dispositivos híbridos y dispositivos móviles.

Este sistema operativo representa una evolución significativa respecto a sus predecesores, introduciendo una serie de características nuevas y mejoradas que mejoran la experiencia del usuario y la productividad. Windows 10 mantiene la familiaridad del escritorio clásico de Windows 7 mientras incorpora elementos modernos de Windows 8, como el menú de inicio rediseñado, que combina la funcionalidad tradicional con los mosaicos dinámicos. (GEEKNETIC, 2024) (Visualícese en la **Tabla 11**. De características ventajas y desventajas).

Tabla 61 Características, ventajas y desventajas de Windows 10

Características	Descripción	Ventajas	desventajas
Interfaz Familiar	Combina la familiaridad de Windows 7 con elementos modernos de Windows 8.	Facilita la transición para usuarios anteriores.	Menos innovador para algunos.
Cortana	Asistente personal digital para búsquedas y comandos de voz.	Interactividad y conveniencia adicionales.	Requiere configuración y acceso a Internet.
Seguridad Mejorada	Incluye Windows Defender y Windows Hello para protección y autenticación.	Protege contra virus y amenazas.	Puede requerir hardware específico.
Actualizaciones Continuas	Modelo de servicio continuo para actualizaciones de características y seguridad.	Siempre tiene acceso a últimas mejoras.	Tiene Interrupciones en el flujo de trabajo.

7. Análisis

7.1. Requerimientos

PINTUMEX, es un sitio web que tiene dos tipos de arquitecturas: microservicios y dos capas. Ya que la arquitectura de microservicios es un método de desarrollo de aplicaciones software que funciona como un conjunto de pequeños servicios que se ejecutan de manera independiente y autónoma, proporcionando una funcionalidad de negocio completa. En ella, cada microservicio es un código que puede estar en un lenguaje de programación diferente, y que desempeña una función específica. Los microservicios se comunican entre sí a través de APIs, y cuentan con sistemas de almacenamiento propios, lo que evita la sobrecarga y caída de la aplicación.

Los microservicios han creado infraestructuras IT más adaptables y flexibles. Porque si se quiere modificar solamente un servicio, no es necesario alterar el resto de la infraestructura. Cada uno de los servicios se puede desplegar y modificar sin que ello afecte a otros servicios o aspectos funcionales de la aplicación.

Mientras que, también es dos capas porque al momento de la implementación del sistema final, este se monta a un servidor, por lo que el cliente solicita recursos y dicho servidor responde directamente a la solicitud, con sus propios recursos. Esto significa que el servidor no requiere otra aplicación para proporcionar parte del servicio.

En este proceso de análisis de requerimientos se trata de capturar y describir detalladamente los requisitos de funcionalidad y de calidad de servicio del producto que se desarrolla según el módulo otorgado, siempre tomando en cuenta los siguientes puntos:

- Un estudio profundo de la necesidad de crear un módulo de registro del distribuidor para automatizar los procesos de seguimiento y control del sistema Pintumex.
- Diseño del módulo que se va a implementar de acuerdo con los requisitos entregados por la empresa, generando documentación en la que se especifica la estructura del software.
- Desarrollo y pruebas del sistema de acuerdo con el diseño realizado, generando entregables y documentación en la que se evidencia la calidad del software.

es crucial especificar no solo qué información se necesita, sino también por qué se estructura de una manera particular. Esto no solo facilita el entendimiento de los requisitos por parte del equipo de desarrollo, sino que también asegura que todos los interesados comprendan las razones detrás de ciertas decisiones de diseño.

7.2 Requerimientos funcionales

Los requerimientos funcionales de un sistema son aquellos que describen las actividades y comportamientos específicos que debe realizar el sistema o software bajo ciertas condiciones. En otras palabras, estos requerimientos detallan cómo debe comportarse el sistema en diferentes escenarios y situaciones para cumplir con sus objetivos operativos.

los actores principales son: administrador y distribuidor prospecto. Cada uno de estos roles tiene funciones y comportamientos específicos dentro del sistema, los cuales se describen a continuación:

- ✓ **Prospecto:** Una vez que un distribuidor prospecto se registra en el sistema, es considerado como prospecto. En esta etapa, el distribuidor ha proporcionado sus datos, pero aún no ha sido verificado ni aprobado para acceder a todas las funciones del sistema.
- ✓ **Administrador:** Es el encargado de revisar y validar los datos proporcionados por los prospectos. Esta validación es crucial para asegurar que la información sea correcta y confiable. Solo después de la aprobación del administrador, el prospecto puede convertirse en distribuidor.
- ✓ **Distribuidor:** Un prospecto que ha sido aprobado por el administrador se convierte en distribuidor. Este rol implica que el usuario tiene privilegios adicionales y responsabilidades dentro del sistema, como acceder a funciones avanzadas o realizar transacciones específicas.

El flujo de trabajo típico se describe de la siguiente manera:

- **Registro de distribuidor prospecto:** Un distribuidor prospecto se registra en el sistema proporcionando la información necesaria.
- **Validación de Prospecto:** El administrador revisa los datos del nuevo registro. Durante este proceso, el administrador puede solicitar información adicional o verificar la autenticidad de los datos proporcionados.
- **Aprobación y Conversión a Distribuidor:** Una vez que el administrador ha validado la información, el prospecto es aprobado y se convierte en distribuidor, obteniendo así acceso completo a las funcionalidades del sistema que le correspondan.
- Este proceso asegura que solo los usuarios verificados y confiables accedan a ciertas funciones críticas del sistema, manteniendo así la integridad y seguridad del mismo. Los requerimientos funcionales detallados para cada uno de estos roles son esenciales para garantizar un flujo de trabajo coherente y seguro dentro del sistema, asegurando que todas las actividades se realicen de manera efectiva.

La **tabla 12** muestra el resumen de los requisitos funcionales del registro del distribuidor prospecto. La tabla incluye 2 columnas para el requerimiento y la descripción. Los requisitos se clasifican en cinco categorías: acceso al portal, captura de datos, validación de campos, envío de solicitud y mensaje de confirmación.

Tabla 12 Requerimientos funcionales del registro del distribuidor prospecto

Requerimiento	Descripción
Acceso al portal	Permitir a los distribuidores prospectos acceder al formulario de registro desde el portal.
Captura de datos	Capturar y almacenar los datos requeridos del distribuidor, incluyendo estado, municipio, código postal, nombre, correo electrónico, teléfono, celular y ubicación del negocio.
Validación de campos	Presentar un aviso de privacidad antes de que el prospecto envíe la solicitud de registro.
Opción de limpiar campos	Ofrecer la opción de limpiar los campos del formulario en caso de que estén mal escritos.
envío de solicitud	Enviar la solicitud de registro al administrador si todos los campos están correctamente llenos.
Mensaje de confirmación	Mostrar una confirmación al usuario una vez que se haya enviado la solicitud de registro.

La **tabla 13** resume los requisitos funcionales de gestión del administrador. Se detallan las tareas que puede realizar para gestionar los registros de los distribuidores prospectos.

Tabla 13 Requerimientos funcionales de gestión del administrador

Requerimiento	Descripción
Acceso a la lista de registros	Permitir al administrador acceder a la lista de distribuidores prospectos registrados.
Modificación de datos	Habilitar al administrador para ver, modificar y agregar comentarios a los datos de los prospectos.
Agregar información interna	Permitir al administrador agregar información de control interno, como CEDIS, tipo de cliente, medio de contacto, referencia, tipo de inversión e inversión.
Agregar observaciones y actividades	Permitir al administrador agregar observaciones y actividades de seguimiento para cada prospecto.
Validación de estatus	Permitir al administrador cambiar el estatus del prospecto según corresponda.
Confirmación de solicitud	El administrador debe tener la capacidad de aprobar o rechazar la solicitud del prospecto, con la opción de agregar comentarios o razones para el rechazo, si corresponde.
Panel de revisión	El sistema debe proporcionar un panel de revisión donde el administrador pueda ver y verificar toda la información y los documentos cargados por el prospecto.
Notificación de revisión	Una vez que el prospecto ha completado su registro y cargado todos los documentos necesarios, el sistema debe notificar al administrador para que inicie el proceso de revisión.

7.3 Requerimientos no funcionales

Los requisitos no funcionales, también conocidos como requisitos de calidad o requisitos de atributos del sistema, describen cómo debe ser el rendimiento, la seguridad, la usabilidad y otros aspectos del sistema o software más allá de su funcionalidad básica, en la que se presenta la **tabla 14** la cual describe los requerimientos no funcionales del sistema a realizar.

Tabla 14 Requerimientos no funcionales

Categoría	Requerimiento no funcional
Seguridad	- Los datos de los distribuidores prospectos deben ser almacenados de manera segura y en cumplimiento con las regularizaciones de privacidad. - Se deben implementar medidas de seguridad para proteger el formulario de registro y la información ingresada por los usuarios.
Usabilidad	- El formulario de registro debe ser fácil de usar y comprender para los usuarios. - La interfaz para la gestión del administrador debe ser intuitiva y eficiente.
Rendimiento	- El sistema debe ser capaz de manejar múltiples solicitudes de registro y gestión de distribuidores prospectos de manera eficiente, sin experimentar retrasos significativos en el tiempo de respuesta.
Disponibilidad	- El sistema debe estar disponible en todo momento para que los distribuidores prospectos puedan acceder al formulario de registro y para que el administrador pueda gestionar estos mismos. - Se deben programar tiempos de inactividad mínimos para el mantenimiento programado.
Escalabilidad	- El sistema debe ser capaz de manejar un crecimiento en el número de registros de distribuidores prospectos sin comprometer su rendimiento y disponibilidad

7.4 Diccionario de datos

La **tabla 15** detalla los diferentes elementos y sus descripciones esenciales para la gestión eficiente de los asesores y distribuidores prospectos en el sistema.

Tabla 15 Diccionario de datos

Identificador	Tipo	Descripción	Ejemplo
Clave del Asesor	Int	Clave única que identifica a cada asesor dentro del sistema.	OG5.nombreApellido
Cedis	Cadena de caracteres	Centros de Distribución que funcionan como almacenes y distribuyen productos e insumos.	CCZAV
Estatus de Actividades	Cadena de caracteres	Estados de las actividades asignadas a los asesores para clientes prospecto.	PENDIENTE
Medios de Contacto	Cadena de caracteres	Medio por el cual el cliente prospecto conoció la empresa Pintumex.	ASESOR

Manejo de Scrum en el Análisis de Requerimientos

El manejo de Scrum se utilizó para organizar y llevar a cabo este proceso de análisis. A continuación, se detallan las fases y actividades realizadas bajo esta metodología ágil.

Sprint Planning:

- **Objetivo:** Definir las tareas y objetivos del sprint específico dedicado al análisis de requerimientos.

Actividades:

- Reunión con la Project manager para priorizar los requerimientos del módulo de registro del distribuidor.
- Desglose de las historias de usuario relacionadas con los requerimientos en tareas manejables.

Daily Stand-up:

- **Objetivo:** Mantener al equipo sincronizado y asegurar el progreso continuo.

Actividades:

- Cada miembro del equipo comparte sus avances, obstáculos y planes para el día.
- Identificación y resolución rápida de problemas relacionados con la captura de requerimientos.

Sprint Review:

- **Objetivo:** Presentar los avances y obtener retroalimentación.

Actividades:

- Presentación de la documentación inicial de requerimientos a los clientes.
- Revisión y ajuste de los requerimientos basados en la retroalimentación obtenida.

Sprint Retrospective:

- **Objetivo:** Reflexionar sobre el sprint completado y mejorar para futuros sprints.

Actividades:

- Evaluación de lo que funcionó bien y qué aspectos necesitan mejora en el proceso de análisis de requerimientos.
- Implementación de cambios para optimizar la captura y documentación de requerimientos.

Estudio de la Necesidad

- **Objetivo:** Justificar la creación del módulo de registro del distribuidor.

Actividades:

- Investigación y análisis de los procesos actuales de seguimiento y control.
- Identificación de ineficiencias y oportunidades de mejora mediante la automatización.

Diseño del Módulo

- **Objetivo:** Establecer la estructura del software según los requisitos entregados.

Actividades:

- Diseño de diagramas de casos de uso y secuencia.
- Documentación de la estructura del módulo en detalle.

Desarrollo y Pruebas

- **Objetivo:** Implementar el diseño y asegurar la calidad del software.

Actividades:

- Desarrollo del módulo basado en los requerimientos documentados.
- Ejecución de pruebas para verificar la funcionalidad y calidad del sistema.
- Generación de entregables y documentación para evidenciar la calidad.
- Análisis de Requerimientos para la Clave del Asesor

La utilización de Scrum en el análisis de requerimientos permitió una organización y ejecución efectiva del proceso, asegurando la captura precisa y detallada de los requerimientos del sistema. La Minuta de trabajo del 9 de enero de 2024, que se presenta en la **Figura 6**, detalla el objetivo de la reunión, el lugar y la fecha, el nombre del proyecto, los asistentes y los puntos a discutir, así como los acuerdos tomados y las conclusiones y fecha de próxima reunión.

MINUTA DE TRABAJO
09/01/2024

Objetivo de la reunión:	* Explicación general del proyecto a realizar. * Asignación de actividades a realizar .
Lugar y Fecha de la reunión:	C. Adolfo López Mateos 131A, Texcacoac, 90806 Santa Ana Chiautempan, Tlax., a 09 de Enero del 2024.
Nombre del Proyecto:	"DESARROLLO DEL MÓDULO REGISTRO DEL DISTRIBUIDOR EN LA PLATAFORMA INTEGRAL PARA SEGUIMIENTO, CONTROL Y MEJORA DE PROCESOS DE PINTUMEX"

ASISTENTES:

Nombre	Puesto
Armando Cervantes Díaz	Desarrollador
Julio Ernesto Pérez Montiel	Desarrollador
Adriana López Tuxpan	Ventas
Israel Pérez Juárez	Desarrollador
Mauricio Aguilar Marín	Desarrollador

1. PUNTOS A DISCUTIR

Explicación general de los requerimientos por parte del cliente.

- ✓ Asignación de actividades a realizar por los equipos de Trabajo.
- ✓ Agendar próxima fecha de reunión con el cliente.
- ✓ Realizar conclusiones y próximos acuerdos.

2. ACUERDOS TOMADOS

Acuerdos
✓ A cada persona se le asignaran actividades. ✓ Seleccionar un encargado por equipo. ✓ Agendar cita con el cliente para la explicación del proyecto. ✓ Dar a conocer en qué consiste el proyecto para empezar a desarrollarlo.

3. CONCLUSIONES

Se asignaron actividades a desarrollar para la elaboración del proyecto como se muestra a continuación:

4. PROXIMA REUNIÓN

Fecha:	Hora:	Lugar:
13 de Enero del 2024	10:00 am	Sala 2

Figura 6 Minuta de trabajo

7.5 Asignación de actividades

La **tabla 16** proporciona una descripción detallada de las actividades realizadas en el marco del proyecto, incluyendo los responsables asignados, los tiempos estimados de ejecución, las fechas de inicio y finalización reales, y las notas explicativas sobre cualquier retraso o contratiempo que se haya presentado. Esta información ha sido fundamental para el seguimiento del progreso del proyecto y la toma de decisiones oportunas para garantizar su éxito.

Tabla 16 Asignación de actividades

No	Referencia	Tipo	Actividades de desarrollo	Persona resuelve	Tiempo	Fecha de inicio	Fecha de terminación	Notas en caso de retraso	Tiempo real
2	1 N/A	contrato	Generación de aplicación móvil para desarrollo de funcionalidades de catálogo de bodegas existentes en el portal almacén	Mauricio	16	04/03/2024	07/03/2024		24
3	2 N/A	contrato	Acceso y consulta de ordenes de traslado en el portal almacén	Julio	8	04/03/2024	05/03/2024		8
4	3 N/A	contrato	Validación y escaneo de cada uno de los productos para validar disponibilidad de equipos para proceso de selección de pedido y	Mauricio	40	15/03/2024		ajuste de información con	0
5	4 N/A	contrato	Generar catálogo de trabajadores del área en el portal almacén		24				0
6	5 N/A	contrato	disponibilidad de equipos para proceso de selección de pedido y	Mauricio	64	15/03/2024		ajuste de información con	0
7	6 N/A	contrato	portal almacén		32				0
8	7 N/A	contrato	Generar portal para visualización y consulta de ordenes de traslado		40				0
9	8 N/A	contrato	Generación de bitácora de información de los productos en el portal		40				0
10	9 N/A	contrato	módulo de consulta de bitácora de producto en el portal	Julio	16	14/05/2024		ajuste de información con	0
11	10 N/A	contrato	nos proporcionen para los prospectos y que mediante un check se	Israel	16	04/03/2024	06/03/2024		16
12	11 N/A	ajuste de	mostrar menús en base a las nuevas calificaciones que se	Israel	12	09/05/2024			0
13	12 N/A	nuevo	administración al portal de clientes, en los módulos de clientes y en el	Joe	24	29/04/2024	04/05/2024		40
14	13 N/A	Bug	administración la carga de múltiples archivos	Israel	12	02/05/2024	03/05/2024		8
15	14 N/A	ajuste de	usuario pueda leerla desde su portal	Sandy	8	15/03/2024		estamos pendientes de la	0
16	15 N/A	modulo/requer	administración que permita que el administrador pueda seleccionar de	Juan	12	04/03/2024		de actividades para terminar	0
17	16 N/A	ajuste de	tomando los correctos, en el portal de administración	Armando	4	06/05/2024	08/05/2024		16
18	17 N/A	Bug	Verificar que los permisos de los menús estén correctamente	Armando	4	06/05/2024	08/05/2024		16
19	18 N/A	Bug							
20	19 N/A	modulo/requer	Creación del módulo de cronograma de apertura tep, en base a las	Mauricio	10	01/02/2024		actividades a realizar	0
21	20 N/A	ajuste de	Agregar/ajustar los filtros en el sistema de asesores donde les permita	Armando	16	16/05/2024	21/05/2024	ende nos atrasara	40
22	21 N/A	ajuste de	Ajustar el módulo de prospectos, donde se le pueda agregar al	Julio	16	07/05/2024	09/05/2024		16

8. Diseño

8.1 Diseño y casos de uso

En este punto, se detallan los diseños de los requerimientos tanto funcionales como no funcionales que cumple la solución diseñada.

Diagramas de casos de uso distribuidor prospecto

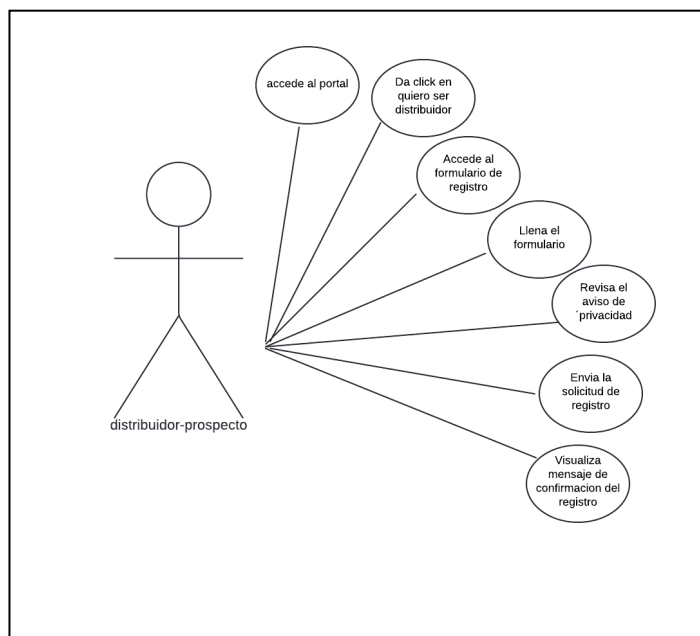


Figura 7 Diagrama de CU – Gestión de distribuidor prospecto

En la **figura 7** se presenta de manera detallada el diagrama de casos de uso para la gestión de distribuidores prospectos. El distribuidor prospecto inicia el proceso de registro al dirigirse a la página de registro del portal, donde selecciona la opción para registrarse como distribuidor.

Al seleccionar esta opción, el distribuidor prospecto es dirigido al formulario de registro diseñado específicamente para distribuidores, donde proporcionará su información personal y comercial. Una vez en el formulario, el usuario completa cuidadosamente todos los campos requeridos, asegurándose de proporcionar información precisa y completa. Antes de enviar su solicitud de registro, el usuario revisa detenidamente el aviso de privacidad proporcionado por el portal, el cual describe cómo se manejará su información personal. Posteriormente se envía su solicitud de registro, iniciando formalmente el proceso de creación de la cuenta del distribuidor en el sistema del portal. Que al mismo tiempo recibe un mensaje de confirmación en pantalla, indicando que su cuenta ha sido creada con éxito.

Diagrama de casos de uso, administrador

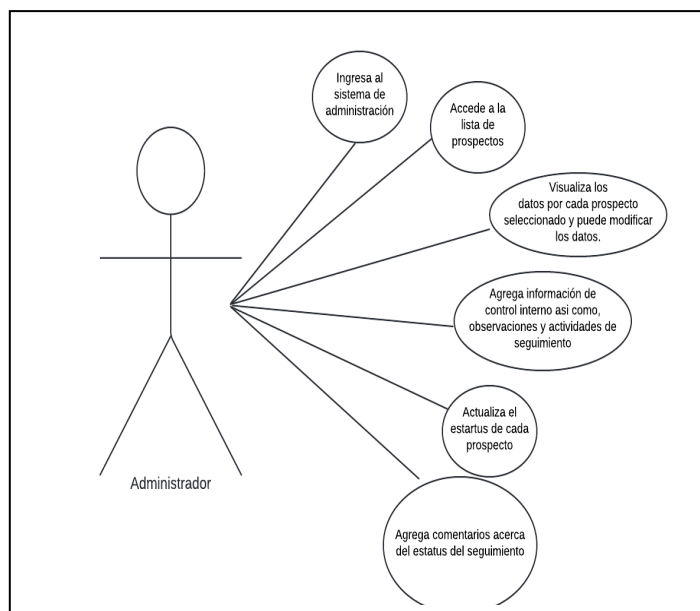


Figura 8 Diagrama de CU – Gestión del administrador

En este apartado, se proporciona una descripción detallada del diagrama de casos de uso de gestión de prospectos por parte del administrador, haciendo referencia a la **figura 8**.

El Administrador tiene la capacidad de iniciar sesión en el sistema de administración utilizando sus credenciales de acceso, lo que le permite acceder a todas las funcionalidades del sistema.

Una vez dentro del sistema, el Administrador puede consultar la lista de distribuidores prospectos disponibles, lo que le proporciona una visión general de todos los prospectos registrados en el sistema.

Para cada prospecto en la lista, el Administrador puede visualizar todos los datos asociados y, si es necesario, realizar modificaciones en estos datos para asegurar su precisión y actualización.

Además de la información básica del prospecto, el Administrador puede agregar información adicional de control interno, así como observaciones y actividades de seguimiento específicas para cada prospecto.

El Administrador también tiene la capacidad de actualizar el estatus de cada prospecto en función de su progreso en el proceso de ventas, lo que ayuda a organizar y priorizar las actividades de seguimiento.

Además de actualizar el estatus del prospecto, el Administrador puede agregar comentarios adicionales acerca del estatus del seguimiento para proporcionar detalles adicionales sobre la interacción con el prospecto.

Estos casos de uso permiten al Administrador gestionar eficazmente los prospectos en el sistema de administración, desde el acceso inicial hasta el seguimiento continuo y la actualización de su estatus y detalles asociados.

8.2 Diagramas de secuencia

Gestión de distribuidores prospectos

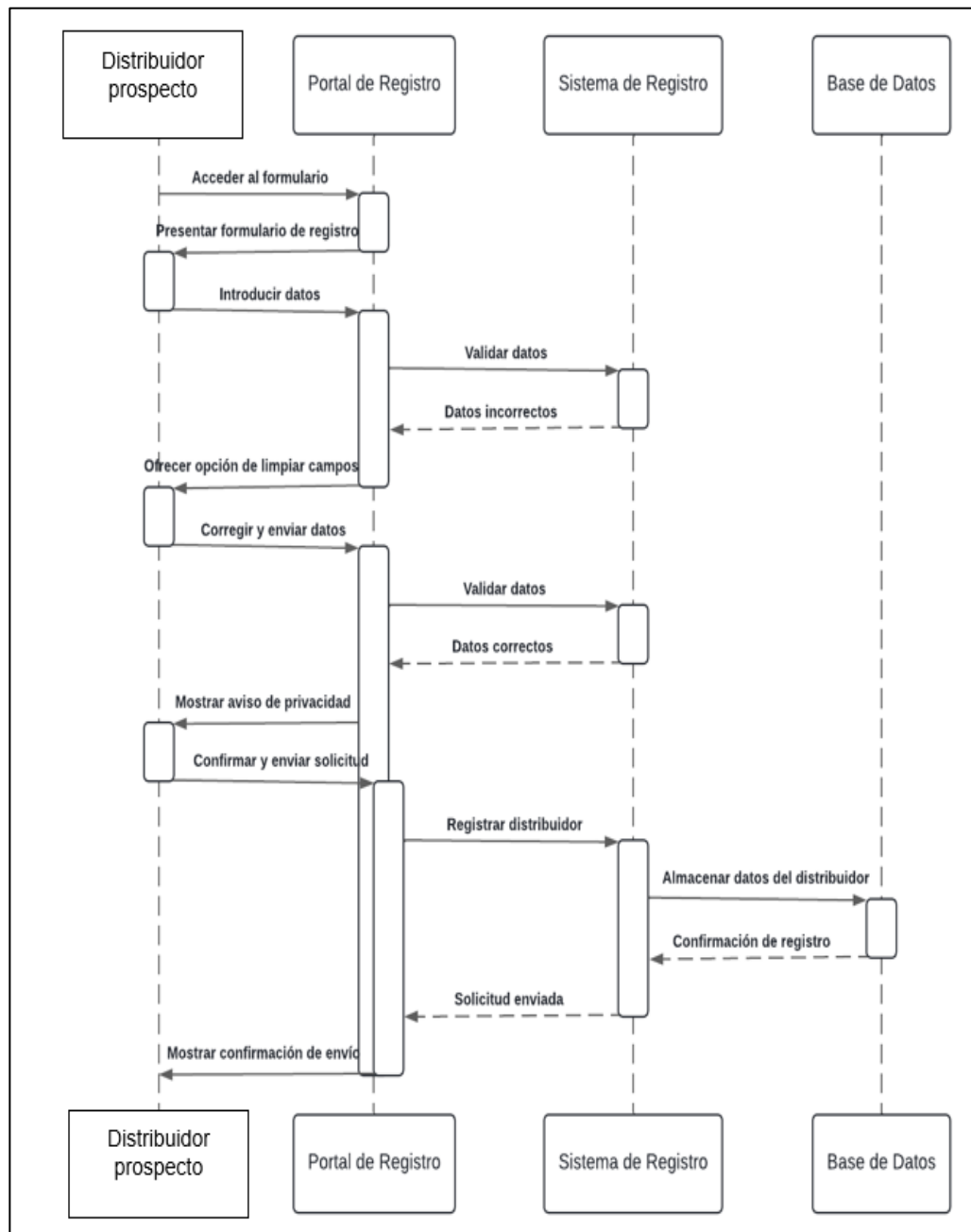


Figura 9 Diagrama de Secuencia – Gestión de distribuidores prospectos

En este apartado, se muestra la descripción detallada del diagrama de la **figura 9**, la cual describe el funcionamiento de la gestión de usuarios para el registro de distribuidores.

El proceso inicia cuando el distribuidor prospecto ingresa al portal de registro con la intención de registrarse y acceder al formulario de registro.

El portal de registro, al detectar la solicitud del usuario, muestra el formulario de registro en la interfaz para que el usuario pueda comenzar a completarlo.

El usuario procede a introducir sus datos personales en los campos correspondientes del formulario.

Luego de que el usuario haya completado el formulario, el sistema de registro se encarga de validar los datos proporcionados.

En caso de que los datos ingresados sean incorrectos, el sistema de registro muestra un mensaje de error al usuario, ofreciendo la opción de limpiar los campos o corregir los datos incorrectos.

El usuario realiza las correcciones necesarias en el formulario y lo envía nuevamente.

El sistema de registro vuelve a validar los datos ingresados por el usuario para asegurarse de su precisión.

Si los datos son correctos, el sistema de registro muestra un aviso de privacidad al usuario, solicitando su confirmación y aceptación.

El usuario lee y acepta el aviso de privacidad, confirmando su consentimiento para enviar la solicitud de registro.

El sistema de registro procede entonces a registrar al distribuidor en la base de datos, almacenando los datos proporcionados por el usuario.

Una vez completado el registro en la base de datos, el sistema envía una confirmación de registro al usuario para informarle que su solicitud ha sido procesada con éxito.

Finalmente, el sistema de registro muestra un mensaje de confirmación de envío al usuario, indicando que su solicitud ha sido enviada correctamente.

Diagrama de secuencia, administrador

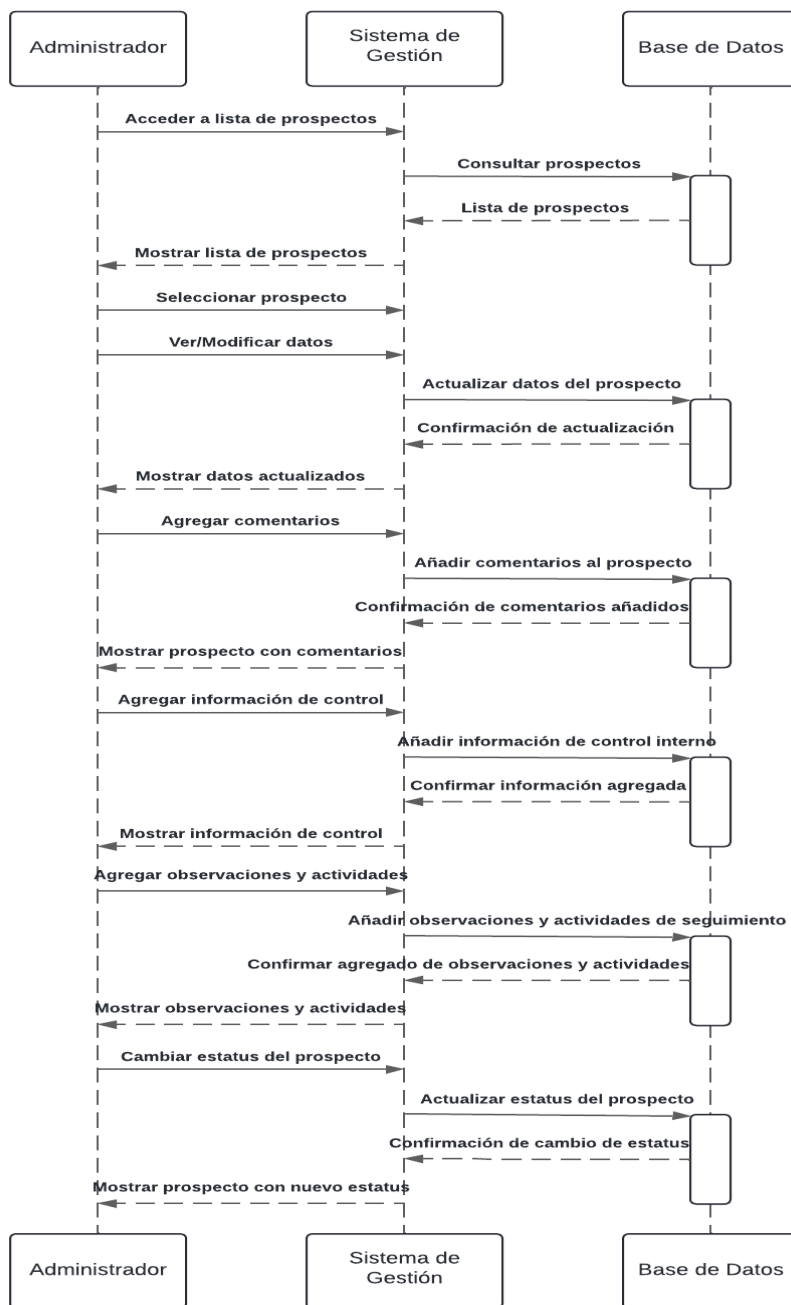


Figura 10 Diagrama de Secuencia – Gestión de administrador

En el contexto del diagrama de secuencia para la gestión de distribuidores prospectos por parte del administrador, se detallan a continuación las interacciones clave, haciendo referencia a la **figura 10**.

El proceso se inicia cuando el Administrador accede al sistema de gestión con el propósito de crear y administrar propuestas.

El Sistema de Gestión, al recibir la solicitud del Administrador, consulta la base de datos para obtener la lista de prospectos disponibles.

Posteriormente, el Sistema de Gestión muestra al Administrador la lista de prospectos obtenida.

El Administrador selecciona un prospecto específico de la lista proporcionada.

Una vez seleccionado, el Administrador tiene la capacidad de visualizar y modificar los datos del prospecto según sea necesario.

Si el Administrador realiza modificaciones en los datos del prospecto, el Sistema de Gestión actualiza la información correspondiente en la base de datos.

Una vez completada la actualización, el Sistema de Gestión muestra una confirmación al Administrador, informándole que los datos del prospecto se han actualizado exitosamente.

Seguidamente, el Sistema de Gestión presenta al Administrador los datos actualizados del prospecto para su revisión.

El Administrador tiene la opción de agregar comentarios adicionales sobre el prospecto. Estos comentarios son incorporados al prospecto por el Sistema de Gestión, y se almacenan en la base de datos.

Una vez añadidos los comentarios, el Sistema de Gestión muestra una confirmación al Administrador, indicando que los comentarios se han agregado satisfactoriamente.

Posteriormente, el Sistema de Gestión muestra al Administrador el prospecto junto con los comentarios recién añadidos.

El Administrador puede agregar información de control interna relacionada con el prospecto.

Dicha información es añadida al prospecto por el Sistema de Gestión y se guarda en la base de datos.

Una vez agregada la información de control, el Sistema de Gestión muestra una confirmación al Administrador, confirmando que la información se ha añadido correctamente.

Seguidamente, el Sistema de Gestión muestra al Administrador la información de control recién agregada del prospecto.

El Administrador puede añadir observaciones y actividades de seguimiento relacionadas con el prospecto.

Estas observaciones y actividades son agregadas al prospecto por el Sistema de Gestión y se almacenan en la base de datos.

Una vez añadidas, el Sistema de Gestión muestra una confirmación al Administrador, indicando que las observaciones y actividades de seguimiento se han añadido correctamente.

Luego, el Sistema de Gestión muestra al Administrador las observaciones y actividades de seguimiento asociadas con el prospecto.

El Administrador tiene la facultad de cambiar el estado del prospecto según sea necesario.

El Sistema de Gestión actualiza el estado del prospecto en la base de datos de acuerdo con la acción realizada por el Administrador.

Una vez completada la actualización del estado, el Sistema de Gestión muestra una confirmación al Administrador, informándole que el estado del prospecto se ha cambiado exitosamente.

Por último, el Sistema de Gestión muestra al Administrador el prospecto con su nuevo estado actualizado.

8.3 Prototipo

La **figura 11** muestra el prototipo de la página de inicio de Pintumex diseñada para promover sus productos y facilitar la navegación del usuario. En la parte superior, se encuentran el logotipo de Pintumex y una celebración de su 50 aniversario, creando una identidad de marca y destacando su experiencia en el mercado. Justo debajo, hay una barra de navegación que permite a los usuarios acceder rápidamente a secciones como inicio, arquitectónico, industrial, repintado automotriz, decorador, innovatorios y contacto. En el centro de la página, un video promocional para atrapar la atención del visitante, proporcionando información visual sobre los productos de Pintumex. A la derecha del video, se encuentran iconos de redes sociales para Facebook, Twitter e Instagram, junto con un botón destacado para que los interesados puedan convertirse en distribuidores, facilitando así la expansión de la red de ventas de la empresa. Este diseño está pensado para ser intuitivo y atractivo, destacando la información clave y facilitando la interacción del usuario con la marca.



Figura 11 Módulo portal de registro

La **figura 12** muestra el prototipo de registro del distribuidor prospecto de Pintumex diseñado para capturar información de manera eficiente y personalizada. A la izquierda, se dispone un formulario con campos para estado, municipio, código postal, nombre, correo electrónico, teléfono, celular y comentarios, que son esenciales para recopilar la información necesaria para contactar y entender mejor al distribuidor prospecto. A la derecha se encuentra un mapa interactivo de Google Maps que permite a los distribuidores prospectos indicar la ubicación exacta de su negocio a iniciar, facilitando la asistencia y logística personalizada. Debajo del formulario, hay dos botones: uno para limpiar los campos y otro para enviar la solicitud, asegurando así que los datos sean correctos antes de enviarlos. Este diseño facilita la captura precisa de datos y mejora la experiencia del usuario al proporcionar una interfaz clara y funcional.

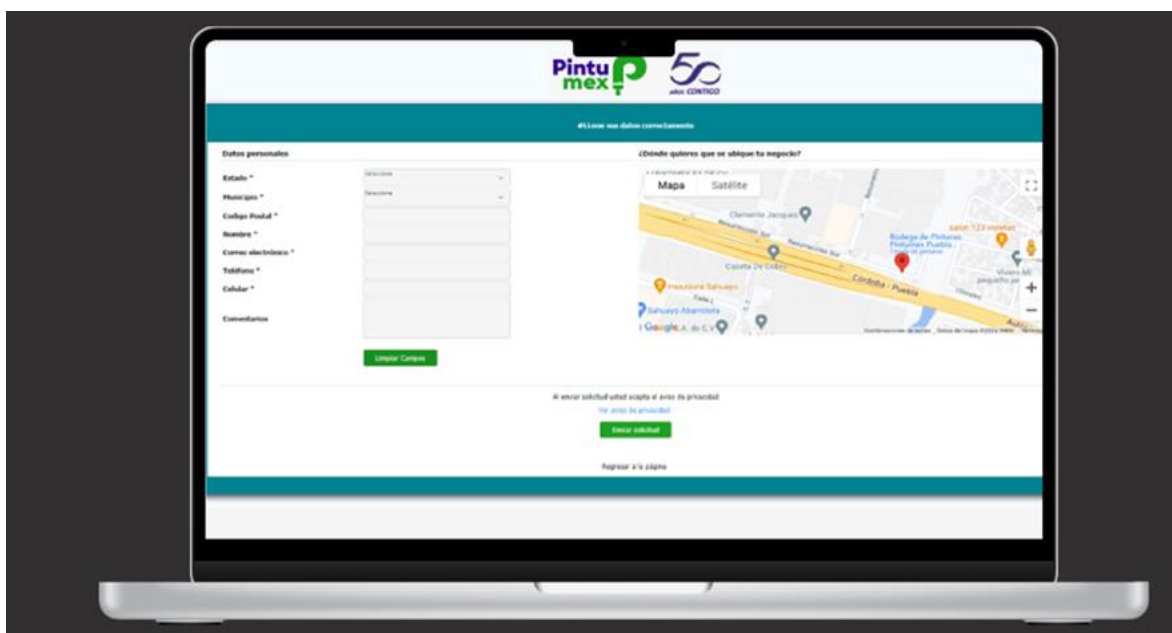


Figura 12 Formulario de registro

La **figura 13** muestra el prototipo de la página de inicio de sesión para el administrador de Pintumex. En el centro de la pantalla, se encuentra un formulario de inicio de sesión sencillo y limpio, donde el administrador debe ingresar su correo electrónico y contraseña

para acceder. El botón "Acceder" está claramente visible para facilitar el acceso. Este diseño está enfocado en la funcionalidad y la simplicidad, asegurando que los administradores puedan iniciar sesión de manera rápida y eficiente.



Figura 13 Inicio de sesión

La **figura 14** muestra el prototipo de bienvenida al sistema de administración. Este diseño ha sido cuidadosamente elaborado para optimizar la experiencia del usuario y mejorar la eficiencia operativa del sistema. En la parte central, se presenta un mensaje de bienvenida acompañado de imágenes y logotipos de Pintumex, incluyendo el eslogan ¡Pinturas que te dan más! Este mensaje centralizado tiene como objetivo proporcionar una entrada amigable y reconocible para los usuarios, reforzando la identidad de la marca y generando confianza desde el primer momento.

En la parte superior, se despliega una barra de navegación con varios menús que forman parte de otros módulos disponibles en el sistema de administración. Estos menús incluyen:

"PROSPECTOS CLIENTES" "CLIENTES" "ASIGNACIONES" "ATENCIÓN A CLIENTES" "CONTENIDOS" "SISTEMA".

La implementación de esta barra de navegación permite un acceso rápido y directo a las diferentes funciones del sistema, facilitando la gestión integral de las operaciones administrativas. Cada módulo está diseñado para abordar aspectos específicos de la administración, como la gestión de clientes, la asignación de tareas, y la atención al cliente, lo que permite a los usuarios realizar sus tareas de manera más organizada y eficiente.



Figura 14 Bienvenida del sistema de administración

La **figura 15** muestra el prototipo de la pantalla de "Prospectos Clientes" del sistema de administración, diseñado para facilitar la gestión y seguimiento de prospectos de distribuidores. La sección principal contiene una tabla detallada que incluye información del registro del prospecto distribuidor, facilitando la comparación y el seguimiento de

The image shows a laptop screen displaying a web application interface for a client management system. The interface is in Spanish and features a navigation bar at the top with tabs: PROSPECTOS CLIENTES, CLIENTES, ASISTENTES, ATENCIÓN A CLIENTES, CONTENIDOS PÓDOL, and SISTEMA. Below the navigation bar, there is a section titled "Programa Clientes" with a "Filtro" dropdown menu set to "Todos".

The main content area displays a table with the following columns: Apodo, Fecha registro, Fecha registro, Nombre, Puesto, Asesor, Estable, Aut. Pto, and Aut. Est. The table lists 9 records, each with a unique ID, a date, a name, a position, an advisor, an establishment, and two authorization status indicators.

At the bottom of the table, there is a summary row showing "Total de registros: 9" and "Págs de cálculo: 1".

La **figura 16** muestra el prototipo de la pantalla de "validación de datos del distribuidor prospecto", diseñado para gestionar y dar seguimiento a distribuidores prospectos. La sección principal incluye campos para modificar y visualizar datos esenciales del prospecto, como nombre, teléfono, correo electrónico, estado, municipio y código postal, además de detalles sobre el origen del prospecto. La sección de control interno registra datos específicos como el CEDIS, medio de contacto, tipo de prospecto y cliente, coordenadas de ubicación y observaciones, con un enlace para ver la ubicación en un

La **figura 16** muestra el prototipo de la pantalla de "validación de datos del distribuidor prospecto", diseñado para gestionar y dar seguimiento a distribuidores prospectos. La sección principal incluye campos para modificar y visualizar datos esenciales del prospecto, como nombre, teléfono, correo electrónico, estado, municipio y código postal, además de detalles sobre el origen del prospecto. La sección de control interno registra datos específicos como el CEDIS, medio de contacto, tipo de prospecto y cliente, coordenadas de ubicación y observaciones, con un enlace para ver la ubicación en un

mapa. A la derecha, la sección de seguimiento permite cambiar la calificación y estado de la visita, agregar comentarios y ver el estado actual del prospecto. En la parte inferior, se encuentra la sección para agregar actividades de seguimiento, facilitando el registro y monitoreo de acciones relacionadas con el prospecto. Este diseño optimiza la gestión de distribuidores prospectos y asegura que toda la información relevante esté accesible para una toma de decisiones informada.

The screenshot displays a web application interface for managing prospects. The interface is divided into several sections:

- Header:** A navigation bar with tabs: PROSPECTOS CLIENTES, CLIENTES, ASIGNACIONES, ATENCIÓN A CLIENTES, CONTENIDOS, and SISTEMA.
- Formulario de registro de prospecto:** A form for entering prospect details. It includes fields for:
 - Folio:** 04240003
 - Nombre:** Jovanny Morales
 - Apellido:** Morales
 - Sexo:** Masculino
 - Edad:** 30
 - Estado:** Nuevo
 - Municipio:** Nuevo
 - Código Postal:** 80000
 - Comentarios:** No
- Información de control interno:** A section for internal control information, including fields for:
 - Código:** 0000
 - Medio de contacto:** Teléfono
 - Tipo de cliente:** Cliente
 - Referencia:** Inversión
 - Latitud:** 19.07738825744
 - Longitud:** -98.280777683672
- Actividades de seguimiento:** A section for tracking activities, with a button labeled "Agregar actividad".
- Seguimiento de prospecto:** A sidebar on the right for tracking the prospect. It includes:
 - Fecha de registro:** 2023/01/01
 - Calificación actual:** PENDIENTE DE CALIFICAR
 - Cambiar Calificación:** A button to change the rating.
 - Cambiar estado:** A dropdown menu with options: ACTIVO, INACTIVO, and SUSPENDIDO.
 - Comentarios estado:** A text area for comments.
 - Estado de prospecto:** A section for the prospect's status.

Figura 16 Validación de datos

9. Desarrollo

9.1 Pasos a implementación

Para el desarrollo del módulo de registro del distribuidor prospecto, se utiliza NetBeans como Entorno de Desarrollo Integrado (IDE) principal para escribir, editar y depurar el código del proyecto. Para ello se configuro con el servidor GlassFish, lo que facilita el despliegue y la prueba de la aplicación en el entorno de desarrollo. La estructura del proyecto está dividida en módulos específicos para administración, asesores, clientes y página web, NetBeans permite la integración y gestión eficiente de dependencias de librerías externas como PrimeFaces y Java Server Faces, (visualice en la **figura 17** de demostración de carga de módulos de pintumex.)

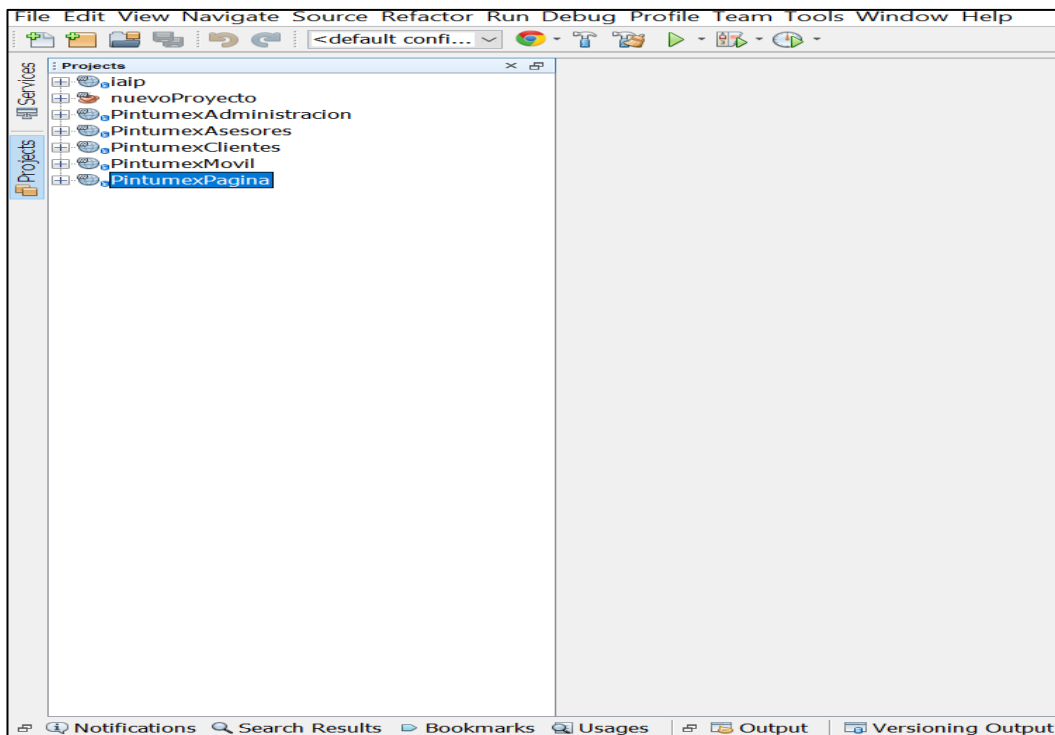


Figura 17 Carga de los Módulos de pintumex

Para la gestión de bases de datos del proyecto, se utilizó Navicat. Este software permitió crear y modificar tablas, vistas y procedimientos almacenados. Así como Establecer una conexión con SQL Server a través de Navicat para administrar los datos utilizados por la aplicación. Además de facilitar la migración de datos entre diferentes entornos y la realización de copias de seguridad, asegurando la integridad y disponibilidad de los datos, como se muestra en la **figura 18**.

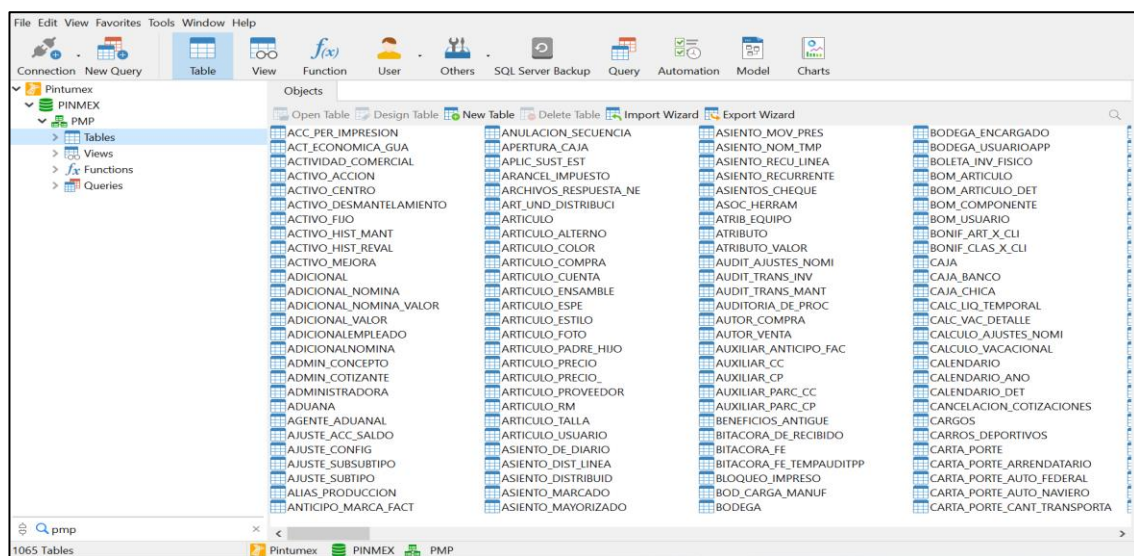


Figura 18 Implementación de la base de datos

TortoiseSVN se utiliza para gestionar el control de versiones del código fuente del proyecto. Esto incluye realizar commits, actualizaciones y merges. TortoiseSVN permite a los miembros del equipo trabajar de manera colaborativa, asegurando que todos estén al tanto de los cambios y facilitando la integración de nuevas funcionalidades. Además, de proporcionar herramientas para gestionar y resolver conflictos cuando múltiples desarrolladores trabajan en el mismo archivo, la **figura 19** muestra un ejemplo de cómo subir cambios desde SVN, los archivos que aparecen en azul son aquellos en los que se hicieron modificaciones en el código y que están listos para subir al proyecto y que los demás miembros del equipo lo puedan tener, la parte de commit se refiere a poner un comentario de lo que se trabajó en esos códigos.

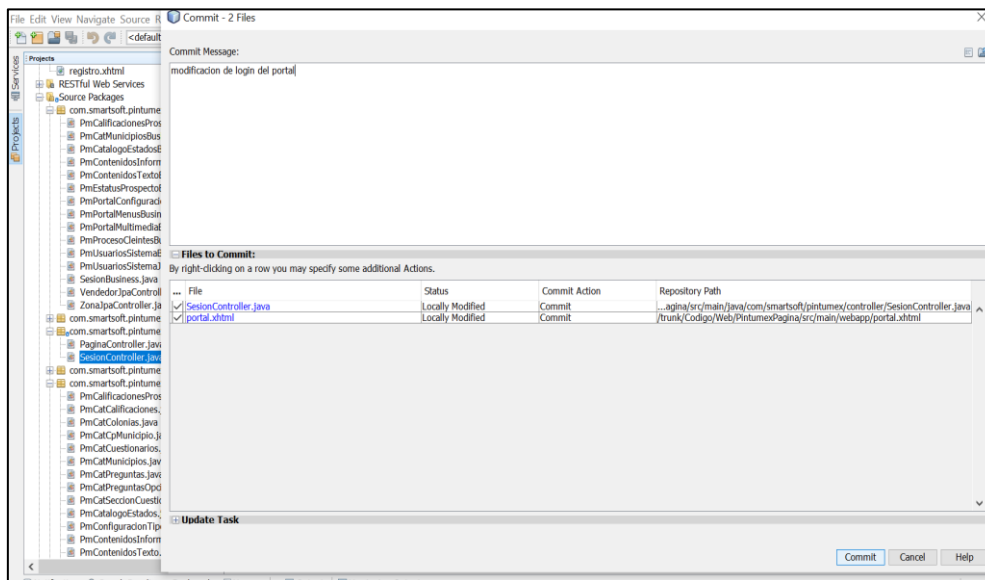


Figura 19 Demostración de subida de cambios usando SVN

9.2 Manejo del lenguaje

Para desarrollar la lógica de negocio del proyecto se utilizó Java como lenguaje principal. Se empleó Managed Beans para gestionar el ciclo de vida de los datos y la interacción con la interfaz de usuario. Así como la Integración de Java Server Faces y PrimeFaces para crear componentes interactivos y dinámicos en la interfaz de usuario. Para el desarrollo del sistema se utilizó la siguiente arquitectura del lenguaje, *xhtml* para definir las vistas, *controller* para manejar la lógica de flujo, *view* para mostrar datos al usuario ya que se utiliza en las páginas *xhtml* para definir la estructura y contenido de la interfaz de usuario, *entity* representa las tablas de la base de datos en forma de objetos Java, facilitando la manipulación y persistencia de datos. Las capas de negocio (*business*) y acceso a datos (*DAO*) se encargan de la lógica empresarial y las operaciones con la base de datos, respectivamente.

```

<div class="EmptyBox10"/>
<p:panelGrid id="idPnelGrid" columns="2" >
  <p:outputLabel value="Folio de registro" rendered="#{sessionController.view.proceso.id!=null}" style="color: #bf0000;font-size: 14px !important;font-weight: bold !important" />
  <p:outputLabel value="#{sessionController.view.proceso.folio}" rendered="#{sessionController.view.proceso.id!=null}" style="color: #bf0000;font-size: 14px !important;font-weight: bold !important" />

  <p:outputLabel value="Nombre" style="font-weight: bold !important"/>
  <p:inputText id="NombreInput" value="#{sessionController.view.proceso.nombre}" style="min-width: 250px" rendered="#{sessionController.view.proceso.id!=null}" ><p:ajax/></p:inputText>
  <p:outputLabel value="#{sessionController.view.proceso.nombre}" rendered="#{sessionController.view.proceso.id!=null}" />

  <p:outputLabel value="Teléfono" style="font-weight: bold !important"/>
  <p:inputNumber id="telefonoInput" thousandSeparator="." decimalPlaces="0" maxValue="9999999999" minValue="0" value="#{sessionController.view.proceso.telefono}" style="min-width: 250px" rendered="#{sessionController.view.proceso.id!=null}" />
  <p:outputLabel value="#{sessionController.view.proceso.telefono}" rendered="#{sessionController.view.proceso.id!=null}" />

  <p:outputLabel value="Celular" style="font-weight: bold !important"/>
  <p:inputNumber id="celularInput" thousandSeparator="." decimalPlaces="0" maxValue="9999999999" minValue="0" value="#{sessionController.view.proceso.celular}" style="min-width: 250px" rendered="#{sessionController.view.proceso.id!=null}" />
  <p:outputLabel value="#{sessionController.view.proceso.celular}" rendered="#{sessionController.view.proceso.id!=null}" />

  <p:outputLabel value="Correo electrónico" style="font-weight: bold !important"/>
  <p:inputText id="correoElectronicoInput" value="#{sessionController.view.proceso.correoElectronico}" style="min-width: 250px" rendered="#{sessionController.view.proceso.id!=null}" />
  <p:outputLabel value="#{sessionController.view.proceso.correoElectronico}" rendered="#{sessionController.view.proceso.id!=null}" />

  <p:outputLabel value="Codigo Postal" style="font-weight: bold !important" />
  <p:inputText id="codigoPostal" value="#{sessionController.view.proceso.codigoPostal}" style="min-width: 250px" rendered="#{sessionController.view.proceso.id!=null}" />
  <p:ajax listener="#{sessionController.cargarDatosPorCodigoPostal()}" update="form" />
  <p:outputLabel value="#{sessionController.view.proceso.codigoPostal}" rendered="#{sessionController.view.proceso.id!=null}" />

  <p:outputLabel value="Municipio" style="font-weight: bold !important"/>
  <p:inputText id="codigoPostal" value="#{sessionController.view.proceso.codigoPostal}" style="min-width: 250px" rendered="#{sessionController.view.proceso.id!=null}" />
  <p:selectOneMenu id="municipioInput" style="min-width: 250px" value="#{sessionController.view.proceso.municipio}" converter="omnifaces.SelectItemsConverter" rendered="#{sessionController.view.proceso.id!=null}" />
  <f:selectItem itemLabel="Seleccione" itemValue="#{null}" />
  <f:selectItems value="#{sessionController.view.listPmCatCpMunicipio}" var="municipio" itemLabel="#{municipio.municipio}" itemValue="#{municipio}" />
  </p:selectOneMenu>

  <p:outputLabel value="Estado" style="font-weight: bold !important"/>

```

Figura 20 Código XHTML del portal

La **Figura 20** muestra un ejemplo del código XHTML para el sitio portal, que sigue las mejores prácticas y estándares de desarrollo. Este código incluye la estructura esencial para un portal, con un formulario separado, un menú de navegación y un área de contenido dinámico, proporcionando una base sólida para el desarrollo de la interfaz del distribuidor prospecto, el código funciona de la siguiente manera.

Comienza con un <div> principal con la clase Container100, que tiene un estilo en línea para establecer un fondo azul, márgenes laterales del 10%, un ancho del 80%, y un borde con sombra para darle un efecto visual elevado. Dentro de este contenedor principal, se utiliza un <p:fragment> para condicionalmente renderizar el contenido.

Dentro de este fragmento, hay otro <div> con fondo blanco y padding, que contiene dos columnas (definidas por la clase Container50). La primera columna contiene etiquetas y campos de entrada para capturar los datos personales del prospecto distribuidor, tales como nombre, teléfono, celular, correo electrónico, código postal, municipio, estado, ciudad y comentarios.

Los campos de entrada están acompañados por etiquetas explicativas y se renderizan condicionalmente dependiendo del estado de *sesionController.view.proceso.id*. Por ejemplo, si *proceso.id* es null, los campos de entrada son editables; de lo contrario, se muestran como etiquetas estáticas.

Cada campo de entrada incluye validaciones y funciones ajax para actualizaciones dinámicas sin recargar la página. En la segunda columna, se incluye un mapa de Google Maps (<p:gmap>) que permite al usuario seleccionar una ubicación específica, la interacción con el mapa es manejada por eventos ajax.

```

34 @Entity
35 @Table(name = "PM_PROCESO_CLIENTES", schema = "PMP")
36 @XmlRootElement
37 @NamedQueries({
38     @NamedQuery(name = "PmProcesoClientes.findAll", query = "SELECT p FROM PmProcesoClientes p WHERE p.fechaAceptacionCliente IS NULL ")
39     , @NamedQuery(name = "PmProcesoClientes.findAllPorEstatus", query = "SELECT p FROM PmProcesoClientes p WHERE p.idEstatus = :estatus AND p.fechaAceptacionCliente IS NULL ")
40     , @NamedQuery(name = "PmProcesoClientes.findById", query = "SELECT p FROM PmProcesoClientes p WHERE p.id = :id")
41     , @NamedQuery(name = "PmProcesoClientes.findByCalificacion", query = "SELECT p FROM PmProcesoClientes p WHERE p.calificacion = :calificacion AND p.fechaAceptacionCliente IS NULL ")
42     , @NamedQuery(name = "PmProcesoClientes.findByCalificacionNull", query = "SELECT p FROM PmProcesoClientes p WHERE p.calificacion is not null AND p.fechaAceptacionCliente IS NULL ")
43     , @NamedQuery(name = "PmProcesoClientes.findByIdCuestionarioRegistro", query = "SELECT p FROM PmProcesoClientes p WHERE p.idCuestionarioRegistro = :idCuestionarioRegistro")
44     , @NamedQuery(name = "PmProcesoClientes.findByIdCuestionarioRegistro", query = "SELECT p FROM PmProcesoClientes p WHERE p.urlCuestionarioRegistro = :urlCuestionarioRegistro")
45     , @NamedQuery(name = "PmProcesoClientes.findByIdCuestionarioAlta", query = "SELECT p FROM PmProcesoClientes p WHERE p.urlCuestionarioAlta = :urlCuestionarioAlta")
46     , @NamedQuery(name = "PmProcesoClientes.findByIdCuestionarioAlta", query = "SELECT p FROM PmProcesoClientes p WHERE p.idCuestionarioAlta = :idCuestionarioAlta")
47     , @NamedQuery(name = "PmProcesoClientes.findByZona", query = "SELECT p FROM PmProcesoClientes p WHERE p.zona = :zona")
48     , @NamedQuery(name = "PmProcesoClientes.findByVendedor", query = "SELECT p FROM PmProcesoClientes p WHERE p.vendedor = :vendedor")
49     , @NamedQuery(name = "PmProcesoClientes.findByIdUsuarioValidoPrimerEtap", query = "SELECT p FROM PmProcesoClientes p WHERE p.idUsuarioValidoPrimerEtap = :idUsuarioValidoPrimerEtap")
50     , @NamedQuery(name = "PmProcesoClientes.findByIdCuestionarioAsesor", query = "SELECT p FROM PmProcesoClientes p WHERE p.idCuestionarioAsesor = :idCuestionarioAsesor")
51     , @NamedQuery(name = "PmProcesoClientes.findByIdUsuarioValidaFinal", query = "SELECT p FROM PmProcesoClientes p WHERE p.idUsuarioValidaFinal = :idUsuarioValidaFinal")
52     , @NamedQuery(name = "PmProcesoClientes.findByName", query = "SELECT p FROM PmProcesoClientes p WHERE p.nombre = :nombre")
53     , @NamedQuery(name = "PmProcesoClientes.findByFolio", query = "SELECT p FROM PmProcesoClientes p WHERE p.folio = :folio"))
54 public class PmProcesoClientes implements Serializable {
55
56     private static final long serialVersionUID = 1L;
57     @Id
58     @Basic(optional = false)
59     @Column(name = "ID")
60     @GeneratedValue(strategy = GenerationType.IDENTITY)
61     private Integer id;
62     @Column(name = "ORIGEN_REGISTRO")
63     private String origenRegistro;
64     @Size(max = 50)
65     @Column(name = "LATITUD_PROSPECTO")
66     private String latitudProspecto;
67     @Size(max = 50)
68     @Column(name = "LONGITUD_PROSPECTO")
69     private String longitudProspecto;

```

Figura 21 Clase entity

La **figura 21** muestra el proceso de creación de la entity (*proceso_Clientes*). Esta entidad es fundamental en el sistema, ya que sirve como la base de datos principal para almacenar información relacionada con los clientes del sistema. Entre los datos que se contienen son: nombre prospecto, dirección, teléfono, celular, correo electrónico, código postal, municipio, estado, ciudad, comentarios, así como longitud y latitud del mapa.

```

31     private PmProcesoClientes proceso;
32
33     public PmUsuariosSistema getUsuario() {
34         return usuario;
35     }
36
37     public void setUsuario(PmUsuariosSistema usuario) {
38         this.usuario = usuario;
39     }
40
41     public PmPortalConfiguraciones getLogo() {
42         return logo;
43     }
44
45     public void setLogo(PmPortalConfiguraciones logo) {
46         this.logo = logo;
47     }
48
49     public PmPortalConfiguraciones getPadingDerecha() {
50         return padingDerecha;
51     }
52
53     public void setPadingDerecha(PmPortalConfiguraciones padingDerecha) {
54         this.padingDerecha = padingDerecha;
55     }
56
57     public PmPortalConfiguraciones getPadingIzquierda() {
58         return padingIzquierda;
59     }
60
61     public void setPadingIzquierda(PmPortalConfiguraciones padingIzquierda) {
62         this.padingIzquierda = padingIzquierda;
63     }
64
65     public boolean getPrevisualizacion() {
66         return previsualizacion;
67     }

```

Figura 22 Clase View

La **figura 22** muestra la creación de la clase view que proporciona una interfaz de usuario intuitiva y funcional en el sistema. Este desempeña un papel crucial al presentar de manera clara y estructurada la información. Al interactuar con el view, se pueden acceder y modificar los datos gracias a los métodos *get* y *set*, que permiten establecer y obtener información. Por ejemplo, los métodos.

```

public PmUsuariosSistema getUsuario() {
    return usuario;
}

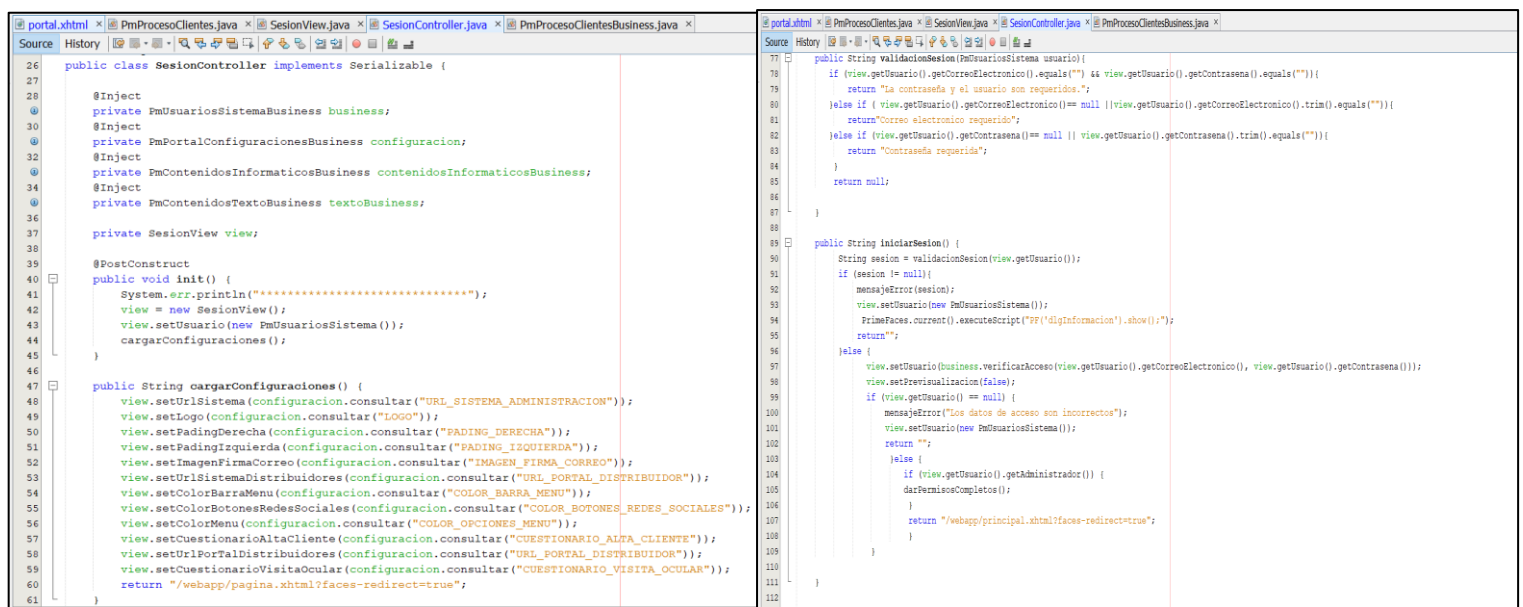
public void setUsuario(PmUsuariosSistema usuario) {
    this.usuario = usuario;
}

```

devuelven el usuario actual y permiten establecer un nuevo usuario con *public void setUsuario(PmUsuariosSistema usuario)*.

Esta clase permite manejar configuraciones visuales como *logo*, *colorMenu*, y *colorBarraMenu*, facilitando la personalización de la interfaz. También permite realizar búsquedas, filtrar resultados y ejecutar otras acciones relevantes para la gestión eficiente de la información.

En resumen, la clase *SesionView* centraliza y gestiona los datos y configuraciones de la sesión, proporcionando una base sólida para la interacción del usuario con la aplicación.



The image shows two side-by-side IDE windows displaying Java code. The left window shows the `SesionController` class, which implements `Serializable`. It contains private fields for various business services and a `SesionView` instance. The `init()` method initializes the view and sets the user. The `cargarConfiguraciones()` method sets various UI configurations like logos, padding, and colors. The right window shows the `SesionView` class, which contains methods for session validation and initialization. The `validacionSesion()` method checks if the user is logged in and if the password is correct. The `iniciarSesion()` method handles the login process, including validation and redirection.

```

26 public class SesionController implements Serializable {
27
28     @Inject
29     private PmUsuariosSistemaBusiness business;
30     @Inject
31     private PmPortalConfiguracionesBusiness configuracion;
32     @Inject
33     private PmContenidosInformaticosBusiness contenidosInformaticosBusiness;
34     @Inject
35     private PmContenidosTextoBusiness textoBusiness;
36
37     private SesionView view;
38
39     @PostConstruct
40     public void init() {
41         System.err.println("*****");
42         view = new SesionView();
43         view.setUsuario(new PmUsuariosSistema());
44         cargarConfiguraciones();
45     }
46
47     public String cargarConfiguraciones() {
48         view.setUrlSistema(configuracion.consultar("URL_SISTEMA_ADMINISTRACION"));
49         view.setLogo(configuracion.consultar("LOGO"));
50         view.setPaddingDerecha(configuracion.consultar("PADING_DERECHA"));
51         view.setPaddingIzquierda(configuracion.consultar("PADING_IZQUIERDA"));
52         view.setImagenFirmaCorreo(configuracion.consultar("IMAGEN_FIRMA_CORREO"));
53         view.setUrlSistemaDistribuidores(configuracion.consultar("URL_PORTAL_DISTRIBUIDOR"));
54         view.setColorBarraMenu(configuracion.consultar("COLOR_BARRA_MENU"));
55         view.setColorBotonesRedesSociales(configuracion.consultar("COLOR_BOTONES_REDES_SOCIALES"));
56         view.setColorMenu(configuracion.consultar("COLOR OPCIONES_MENU"));
57         view.setCuestionarioAltaCliente(configuracion.consultar("CUESTIONARIO_ALTA_CLIENTE"));
58         view.setUrlPortalDistribuidores(configuracion.consultar("URL_PORTAL_DISTRIBUIDOR"));
59         view.setCuestionarioVisitaOcular(configuracion.consultar("CUESTIONARIO_VISITA_OCULAR"));
60         return "/webapp/pagina.xhtml?faces-redirect=true";
61     }
62 }

```

```

77 public String validacionSesion(PmUsuariosSistema usuario) {
78     if (view.getUsuario().getCorreoElectronico().equals("") || view.getUsuario().getContraseña().equals("")) {
79         return "La contraseña y el usuario son requeridos.";
80     } else if (view.getUsuario().getCorreoElectronico() == null || view.getUsuario().getCorreoElectronico().trim().equals("")) {
81         return "Correo electronico requerido";
82     } else if (view.getUsuario().getContraseña() == null || view.getUsuario().getContraseña().trim().equals("")) {
83         return "Contraseña requerida";
84     }
85     return null;
86 }
87
88
89 public String iniciarSesion() {
90     String sesion = validacionSesion(view.getUsuario());
91     if (sesion != null) {
92         mensajeError(sesion);
93         view.setUsuario(new PmUsuariosSistema());
94         PrimeFaces.current().executeScript("PF('dlgInformacion').show()");
95         return "";
96     } else {
97         view.setUsuario(business.verificarAcceso(view.getUsuario().getCorreoElectronico(), view.getUsuario().getContraseña()));
98         view.setPrevisualizacion(false);
99         if (view.getUsuario() == null) {
100             mensajeError("Los datos de acceso son incorrectos");
101             view.setUsuario(new PmUsuariosSistema());
102             return "";
103         } else {
104             if (view.getUsuario().getAdministrador() {
105                 darPermisosCompletos();
106                 return "/webapp/principal.xhtml?faces-redirect=true";
107             }
108         }
109     }
110 }
111
112 }

```

Figura 23 Clase controller

La **figura 23** muestra la creación del controller, que desempeña un papel fundamental en la gestión del flujo de datos y la interacción entre la interfaz de usuario y la lógica de negocio en el sistema. Este controller, se encarga de dirigir las acciones del usuario y coordinar la comunicación con otras capas del sistema. Al interactuar con el controller, los eventos del usuario, como clics de botón o entradas de formulario, se gestionan de manera adecuada y se traducen en acciones concretas dentro del sistema.

El *SesionController* es un ejemplo de un controlador de sesión que cumple las funciones en la plataforma del administrador cuando inicie sesión.

Está definido como un bean gestionado de alcance de sesión mediante las anotaciones `@Named` y `@SessionScoped`, asegura que una instancia del controlador se mantenga durante toda la sesión del usuario.

```
@Named(value = "sesionController")
@SessionScoped
public class SesionController implements Serializable {
```

Utiliza la anotación `@Inject` para inyectar dependencias de negocio ejemplo.

```
@Inject
private PmUsuariosSistemaBusiness business;
@Inject
private PmPortalConfiguracionesBusiness configuracion;
@Inject
private PmContenidosInformaticosBusiness contenidosInformaticosBusiness;
@Inject
private PmContenidosTextoBusiness textoBusiness;
```

permitiendo al controlador interactuar con la base de datos y manejar diferentes aspectos de la lógica de negocio. El controlador contiene una instancia de `SesionView`, que representa la vista asociada a la sesión, y se inicializa en el método `@PostConstruct`, que se llama automáticamente después de la construcción del bean.

```
@PostConstruct
public void init() {
    view = new SesionView();
    view.setUsuario(new PmUsuariosSistema());
    cargarConfiguraciones();
}
```

Este método también llama a *cargarConfiguraciones* para establecer varias configuraciones del sistema en la vista.

El método *iniciarSesion* que se muestra en la parte de abajo, maneja la lógica de inicio de sesión del usuario, incluyendo la validación de las credenciales y la configuración de permisos si el usuario es administrador.

```
public String iniciarSesion() {
    String sesion = validacionSesion(view.getUsuario());
    if (sesion != null){
        mensajeError(sesion);
        view.setUsuario(new PmUsuariosSistema());
        PrimeFaces.current().executeScript("PF('dlgInformacion').show();");
        return "";
    }else {
        view.setUsuario(business.verificarAcceso(view.getUsuario().getCorreoElectronico(),
        view.getUsuario().getContraseña()));
        view.setPrevisualizacion(false);
        if (view.getUsuario() == null) {
            mensajeError("Los datos de acceso son incorrectos");
            view.setUsuario(new PmUsuariosSistema());
            return "";
        }else {
            if (view.getUsuario().getAdministrador())
                darPermisosCompleto();
        }
        return "/webapp/principal.xhtml?faces-redirect=true";
    }
}
```

Aquí el código valida si las credenciales son incorrectas, muestra un mensaje de error y reinicia la instancia del usuario en la vista. Además de que, el controlador incluye métodos para *cerrar sesión* (*cerrarSesion*), *mostrar mensajes de información y error* (*mensajeInformativo*, *mensajeError*), y *cargar textos informativos* (*cargarTexto*).

También maneja la carga de archivos, como logos, y la actualización de configuraciones del sistema.

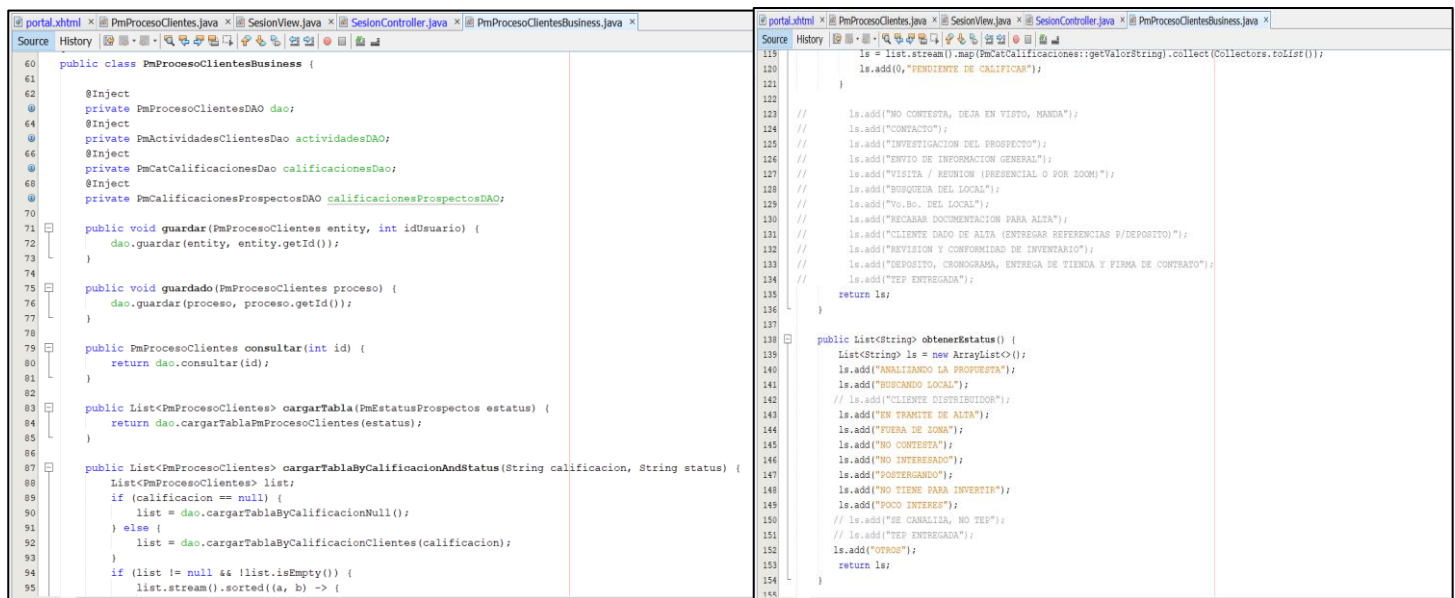


Figura 24 Clase business

La **figura 24** representa la creación del componente business, esencial para el desarrollo y la implementación de la lógica empresarial en el sistema. Este business, actúa como el núcleo de la aplicación, donde se definen y ejecutan las operaciones empresariales clave. Un ejemplo claro de esta implementación es la clase *ProcesoClientesBusiness*, que maneja la lógica de negocio relacionada con la gestión de *clientes(prospectos distribuidores)*. Esta clase encapsula la validación de datos, la implementación de reglas de negocio y la coordinación de operaciones entre diferentes partes del sistema a través de varios métodos.

Guardar y Consultar Registros: La clase incluye métodos como `guardar(PmProcesoClientes entity, int idUsuario)` y `consultar(int id)`, que se encargan de almacenar y recuperar información de los procesos de clientes en la base de datos.

Esto asegura que los datos sean correctamente gestionados y estén disponibles cuando se necesiten.

```
public void guardar(PmProcesoClientes entity, int idUsuario) {  
    dao.guardar(entity, entity.getId());  
}  
public PmProcesoClientes consultar(int id) {  
    return dao.consultar(id);  
}
```

Carga y Filtrado de Información: Los métodos *cargarTabla(EstatusProspectos estatus)* y *cargarTablaByCalificacionAndStatus(String calificación, String status)* permiten cargar listas de procesos de clientes, filtradas por estatus, calificación y estado. Esto facilita la organización y presentación de información relevante para los usuarios del sistema.

```
public List<PmProcesoClientes> cargarTabla(PmEstatusProspectos estatus) {  
    return dao.cargarTablaPmProcesoClientes(estatus);  
}  
public List<PmProcesoClientes> cargarTablaByCalificacionAndStatus(String calificacion, String status) {  
    List<PmProcesoClientes> list;  
    if (calificacion == null) {  
        list = dao.cargarTablaByCalificacionNull();  
    } else {  
        list = dao.cargarTablaByCalificacionClientes(calificacion);  
    }  
    if (list != null && !list.isEmpty()) {  
        list.stream().sorted((a, b) -> {  
            return a.getFechaRegistroPagina().compareTo(b.getFechaRegistroPagina());  
        });  
    }  
    return list;  
}
```

Implementación de Reglas de Negocio: A través del método *contarEstatusActividadesDeProceso(PmProcesoClientes proceso)*, la clase actualiza el número de actividades pendientes y no visualizadas de un proceso. Esto garantiza que las reglas de negocio se apliquen correctamente, manteniendo la integridad y actualidad de los datos.

```
public void contarEstatusActividadesDeProceso(PmProcesoClientes proceso) {
    proceso.setActividadesPendientes(actividadesDAO.contarActividadesPendientes(proceso) + 1);
    proceso.setActividadesSinVisualizar(actividadesDAO.contarActividadesSinVisualizar(proceso));
    guardar(proceso, proceso.getId());
}
```

Validación de Datos: El método *validarRegistroCliente(Nit entityNit, Cliente entityCliente, PmProcesoClientesView view)* realiza una serie de comprobaciones para asegurar que toda la información requerida de un cliente esté completa antes de permitir su registro en el sistema. Esto garantiza la calidad y consistencia de los datos ingresados.

```
public String validarRegistroCliente(Nit entityNit, Cliente entityCliente, PmProcesoClientesView view) {
    if (entityNit.getNit() == null || entityNit.getNit().trim().isEmpty()) {
        return "En el apartado de 'Información del RFC del prospecto recopilada' el campo rfc es requerido";
    }
    if (entityNit.getRazonSocial() == null || entityNit.getRazonSocial().trim().isEmpty()) {
        return "En el apartado de 'Información del RFC del prospecto recopilada' el campo razonSocial es requerido";
    }
    if (entityNit.getAlias() == null || entityNit.getAlias().trim().isEmpty()) {
        return "En el apartado de 'Información del RFC del prospecto recopilada' el campo Alias es requerido";
    }
}
```

Coordinación de Operaciones: Métodos como *obtenerCalificaciones()* y *obtenerEstatus()* devuelven listas de calificaciones y estados posibles para los clientes, facilitando la coordinación entre diferentes partes del sistema que utilizan esta información. Además el método, *obtenerAsesoresConProspectos()* compila una lista de vendedores con sus respectivos prospectos, coordinando la información entre los datos de clientes y los responsables de su seguimiento.

```
public List<String> obtenerCalificaciones() {
    List<String> ls = new ArrayList<>();
    List<PmCatCalificaciones> list = calificacionesDao.cargarTablaCaliicaciones();
    if (list != null) {
        ls = list.stream().map(PmCatCalificaciones::getValorString).collect(Collectors.toList());
        ls.add(0, "PENDIENTE DE CALIFICAR");
    }

    public List<Vendedor> obtenerAsesoresConProspectos() {
        List<PmProcesoClientes> list = dao.obtenerListaCompleta();
        List<Vendedor> listado = list.stream().map(PmProcesoClientes::getVendedor).filter(x -> x != null).distinct().collect(Collectors.toList());
        System.out.println("Lista de vendedores " + listado);
        for (Vendedor vendedor : listado) {
            vendedor.setListProspectos(new ArrayList<>());
            for (PmProcesoClientes prospecto : list) {
                if (prospecto.getVendedor() != null && prospecto.getVendedor().equals(vendedor)) {
                    vendedor.getListProspectos().add(prospecto);
                }
            }
            System.out.println("Vendedor asesor " + vendedor.getNombre() + " tiene estos asesore " + vendedor.getListProspectos());
        }
        return listado;

    public List<String> obtenerEstatus() {
        List<String> ls = new ArrayList<>();
        ls.add("ANALIZANDO LA PROPUESTA");
        ls.add("BUSCANDO LOCAL");
    }
}
```

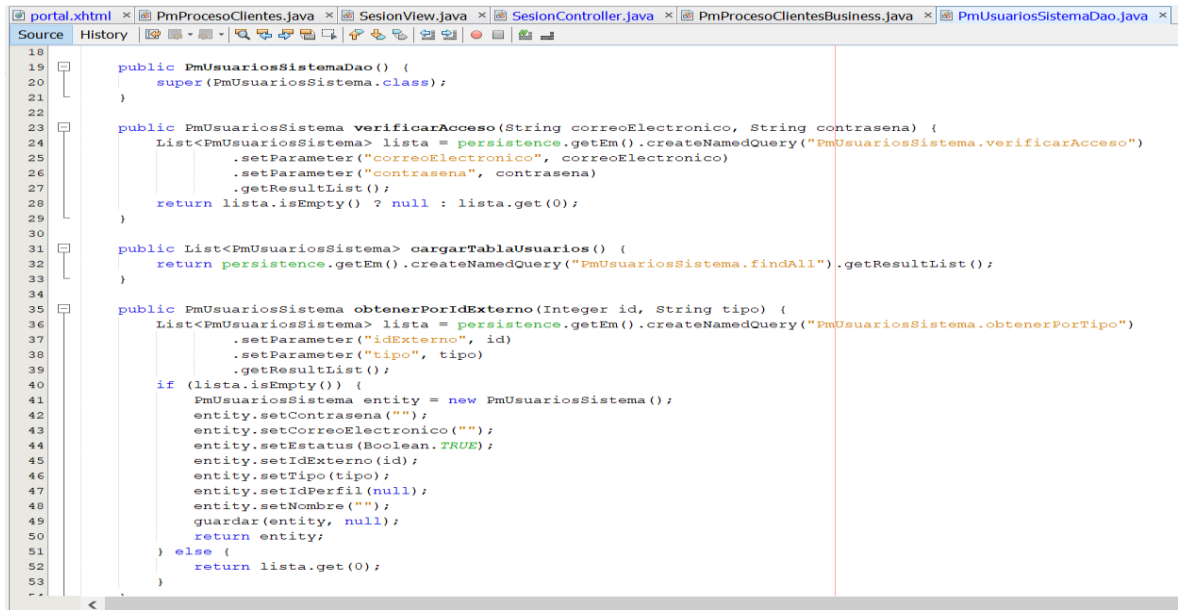


Figura 25 Clase DAO

La **figura 25** muestra la creación del DAO que se utilizó para gestionar el acceso a la base de datos en el contexto de la gestión de clientes. Este componente es esencial para realizar operaciones de creación, lectura, actualización y eliminación de datos. Además, permite la ejecución de consultas personalizadas para obtener información específica según ciertos criterios, como se muestra en el código de abajo mediante el método *verificarAcceso*.

```

public PmUsuariosSistema verificarAcceso(String correoElectronico, String contrasena) {
    List<PmUsuariosSistema> lista =
    persistence.getEm().createNamedQuery("PmUsuariosSistema.verificarAcceso")
        .setParameter("correoElectronico", correoElectronico)
        .setParameter("contrasena", contrasena)
        .getResultList();
    return lista.isEmpty() ? null : lista.get(0);
}

```

El método *verificarAcceso* del DAO recibe un correo electrónico y una contraseña como parámetros, y utiliza el *EntityManager* para ejecutar una consulta nombrada ("*PmUsuariosSistema.verificarAcceso*") que busca en la base de datos un usuario con las credenciales proporcionadas. La consulta establece los parámetros de correo electrónico y contraseña, y luego obtiene una lista de resultados. Si la lista está vacía, el método devuelve null, indicando que no se encontró un usuario con esas credenciales; si la lista no está vacía, devuelve el primer objeto *PmUsuariosSistema* de la lista, que representa al usuario autenticado.

10. Resultados

En este apartado consiste en mostrar los resultados de la plataforma PINTUMEX de acuerdo con los análisis de los requerimientos ya autorizados, permitiendo visualizar ideas y conceptos en el contexto del diseño y elementos en detalle, haciendo uso de la herramienta de Netbeans para la creación de Front-end así como Back-end.

10.1 Portal de distribuidores prospectos

La **figura 26** muestra el portal donde los distribuidores prospectos pueden visualizar un video, las redes sociales de Pintumex e información relevante de la empresa. Este módulo está compuesto por otros módulos más como: **Arquitectónico**, que presenta productos y soluciones de pintura para aplicaciones arquitectónicas y decorativas en construcciones residenciales y comerciales; **Industrial**, que ofrece información sobre recubrimientos y pinturas diseñadas para aplicaciones industriales, incluyendo protección contra corrosión y acabados especializados; **Repintado Automotriz**, que proporciona detalles sobre productos específicos para la industria automotriz, enfocados en el repintado y reparación de vehículos; **Decorador**, que muestra ideas, tendencias y productos para proyectos de decoración interior y exterior; **Nosotros**, que contiene información sobre la historia, misión, visión y valores de la empresa Pintumex; y **Contacto**, que ofrece opciones para que los usuarios se pongan en contacto con la empresa, incluyendo formularios de contacto, números de teléfono y direcciones de correo electrónico. Si el prospecto desea convertirse en distribuidor, da clic en el botón “Quiero ser distribuidor”.



Figura 26 Módulo de portal de registro

10.2 Registro del distribuidor prospecto

La **figura 27** muestra el formulario de registro para el distribuidor prospecto en el cual ingresa su nombre, teléfono, celular, correo electrónico, código postal, municipio, estado, ciudad y comentarios, así como la ubicación de negocio, al llenar todos estos campos al distribuidor prospecto se le mostrara un aviso de privacidad que deberá aceptar para poder enviar la solicitud, para que posteriormente el administrador de pintumex realice su siguiente proceso como administrador.

Figura 27 Formulario de registro

10.3 Inicio de sesión

La **figura 28** muestra el inicio de sesión del administrador de pintumex el cual debe acceder con su correo electrónico y contraseña a la plataforma para visualizar la lista de distribuidores prospectos registrados.

Figura 28 Módulo de inicio de sesión del administrador

10.4 Bienvenida al sistema de administración

La **figura 29** muestra que, al ingresar el administrador con su correo electrónico y contraseña, el sistema despliega una pantalla de bienvenida que incluye los logos de la empresa Pintumex y un mensaje de bienvenida, junto con varios módulos importantes para la gestión del sistema, entre los cuales se encuentran los siguientes: **Prospectos Clientes**, muestra la información de los prospectos registrados a través del formulario del portal; **Clientes**, contiene datos detallados de los clientes actuales, incluyendo historial de compras y datos de contacto; **Asesores**, proporciona información sobre los asesores de ventas y permite el seguimiento de su desempeño; **Atención a Clientes**, gestiona las solicitudes y consultas de los clientes; **Contenidos Públicos**, administra la información y materiales publicados en el portal, como noticias y actualizaciones de productos; **Reportes**, ofrece herramientas para generar reportes sobre diversas actividades y métricas del sistema; y **Sistema**, que permite la configuración y administración del sistema, incluyendo la gestión de usuarios y permisos.



Figura 29 Bienvenida al sistema de pintumex

10.5 Proceso de clientes

La **figura 30** muestra el proceso de los registros de los distribuidores prospectos, es decir, en que estatus se encuentran, ya sea validado, en espera, etc. Así como la fecha del registro, el origen de registro, el cedís asignado, el asesor asignado, el nombre del prospecto, el estado en el que se encuentra el distribuidor prospecto, el estatus y las actividades pendientes por asignar, datos que el administrador tendrá que verificar en el módulo de validación de datos.

PROSPECTOS CLIENTES CLIENTES ASESORES ATENCIÓN A CLIENTES CONTENIDOS PÚBLICOS REPORTES SISTEMA										
Proceso Clientes										
Calificación	TODOS Registrar nuevo prospecto									
Calificación	Folio	Fecha registro	Origen registro	CEDIS	Asesor	Nombre del prospecto	Estado	Estatus	Act. Pen	
4	05240013	27/05/2024	PÁGINA WEB	TEPACATZ	-	Karen Juarez Ramires	Tlaxcala	EN TRAMITE DE ALTA	-1	Q
PENDIENTE DE C	05240012	24/05/2024	ROCKET	LOC	-	Emiliana Hernandez	Tlaxcala	POSTERGANDO	0	Q
PENDIENTE DE C	05240011	24/05/2024	PÁGINA WEB	ACA	-	Brandon Hernandez Juarez	Tlaxcala	BUSCANDO LOCAL	0	Q
PENDIENTE DE C	05240010	23/05/2024	ASESOR	CCZAV	ACS-ALBERTO CASTRO PIÑON	Misael Perez Hernandez	Tlaxcala	ANALIZANDO LA PROPUESTA	0	Q
PENDIENTE DE C	05240009	23/05/2024	PINTUMEX		-	Brandon Hernandez Juarez	Tlaxcala	ESTATUS PENDIENTE	0	Q
PENDIENTE DE C	05240008	23/05/2024	PÁGINA WEB		-	h	Veracruz	ESTATUS PENDIENTE	0	Q
PENDIENTE DE C	05240007	22/05/2024	PÁGINA WEB		-	anahi gomez	Tlaxcala	ESTATUS PENDIENTE	1	Q
PENDIENTE DE C	05240006	22/05/2024	PÁGINA WEB		-	Julio Ernesto Montiel Perez	Aguascalientes	ESTATUS PENDIENTE	0	Q
PENDIENTE DE C	05240005	22/05/2024	PÁGINA WEB		-	brenda morales	Tlaxcala	ESTATUS PENDIENTE	0	Q
PENDIENTE DE C	05240004	21/05/2024	PINTUMEX		-	Julio Ernesto Montiel Perez	Baja California	ESTATUS PENDIENTE	0	Q
PENDIENTE DE C	05240003	17/05/2024	PINTUMEX		-	JOSE ARMINAOD	Aguascalientes	ESTATUS PENDIENTE	0	Q
PENDIENTE DE C	05240003	17/05/2024	PINTUMEX		-	JOSE ARMINAOD	Baja California	ESTATUS PENDIENTE	0	Q
PENDIENTE DE C	05240001	09/05/2024	PÁGINA WEB		OGS-OSCAR GONZALEZ ARRIETA	joe	Tlaxcala	ESTATUS PENDIENTE	0	Q
SIN CALIFICAR	04240001	05/04/2024	PÁGINA WEB		-	jose armando	Baja California	FUERA DE ZONA	0	Q
DEPOSITO, CROI	11230008	29/11/2023	PÁGINA WEB		ACS-ALBERTO CASTRO PIÑON	Jose Aguilar Gonzalez	Puebla	ANALIZANDO LA PROPUESTA	0	Q

Figura 30 Pantalla de proceso de clientes

10.6 validación de datos

La **figura 31** muestra la validación de los datos del distribuidor prospecto, donde el administrador verifica que los datos del formulario de registro sean correctos en caso contrario que estén mal este podrá hacer modificaciones, posteriormente podrá cambiar el estatus a validar e ingresar información interna para tener un mayor control del distribuidor prospecto, así como también agregar actividades de filtro de seguridad, poner observaciones y calificar el seguimiento del distribuidor prospecto, por último el administrador puede guardar los cambios.

PROSPECTOS CLIENTES | **CLIENTES** | **ASESORES** | **ATENCIÓN A CLIENTES** | **CONTENIDOS PÚBLICOS** | **REPORTES** | **SISTEMA**

Proceso Clientes

Información de registro de prospecto

Folio 05240013

Nombre: Karen Juarez Ramirez

Teléfono: 2418979888

Estado: Tlaxcala

Ciudad: Calpulalpan

Comentarios: Quiero ser distribuidor para la parte de calpulalpan

Origen: PÁGINA WEB

Correo electrónico: Karen20@gmail.com

Celular: 2468979789

Municipio: Calpulalpan

Código Postal: 90200

Información de control interna

CEDIS: TEPACATZ - TEP PINTUTODO ACATZINGO

Medio de contacto: Página

Tipo de cliente: Distribuidor

Monto de inversión: 200,000.00

Información de ubicación

Latitud: 19.0752841242281

Longitud: -98.14184434853814

Seguimiento de prospecto

Fecha de registro: 27/05/2024

Calificación actual: VISITA / REUNION (PRESENCIAL O POR ZOOM)

Cambiar Calificación

Cambiar estatus: EN TRAMITE DE ALTA

Observaciones: se revisara la información

Estatus de prospecto:

Actividades de filtro de seguridad

Agregar actividad

B

Administrador

27/05/2024 10:31:53 se realizara una visita para ver el espacio y ver si puede ser aprobado como distribuidor

27/05/2024

Figura 31 Validación de datos

11. Pruebas unitarias

```
private MunicipioDao municipioDao;

@Before
public void setUp() {
    MockitoAnnotations.initMocks(this);
}

@Test
public void testFindByCodigoPostal() {
    PmCatCpMunicipio municipio = new PmCatCpMunicipio();
    municipio.setCodigoPostal(12345);

    when(municipioDao.findByCodigoPostal(12345)).thenReturn(municipio);

    PmCatCpMunicipio result = municipioService.findByCodigoPostal("12345");
    assertNotNull(result);
    assertEquals(12345, result.getCodigoPostal());
}

@Test
```

Figura 32 Prueba de validación del comportamiento del método *findByCodigoPostal*

La **figura 32** muestra una prueba de validación del método *findByCodigoPostal* de la clase *Municipio*. El objetivo es verificar que este método funcione correctamente, devolviendo un objeto *CatCpMunicipio* que represente el municipio correspondiente al código postal especificado. En la prueba, se crea un nuevo objeto *CatCpMunicipio* y se le asigna el código postal "12345". Se utiliza la biblioteca Mockito para crear un simulacro del objeto *municipioDao*, configurándolo para devolver el objeto creado cuando se llama al método con el código postal "12345". Luego, se invoca el método *findByCodigoPostal* del objeto *Municipio* y se verifica que el objeto devuelto no sea nulo.

```

@Test
public void testRegistrarClienteValid() {
    Cliente cliente = new Cliente();
    cliente.setNombre("Nombre Cliente");
    cliente.setTelefono("1234567890");
    cliente.setCelular("0987654321");
    cliente.setCorreoElectronico("cliente@example.com");
    cliente.setCodigoPostal("12345");
    cliente.setMunicipio("Municipio");
    cliente.setEstado("Estado");
    cliente.setCiudad("Ciudad");

    boolean result = municipioService.registrarCliente(cliente);
    assertTrue(result);
}

```

Figura 33 Prueba de validación de campos

La **figura 33** muestra una prueba unitaria que valida el registro de un distribuidor prospecto mediante el método *registrarCliente*. Esta con la finalidad de verificar los campos para garantizar que todos los campos requeridos (nombre, teléfono, celular, correo electrónico, código postal, municipio, estado) sean validados correctamente.

```

@Mock
private MapaDao mapaDao;

@Before
public void setUp() {
    MockitoAnnotations.initMocks(this);
}

@Test
public void testGuardarUbicacion() {
    double latitud = 19.4326;
    double longitud = -99.1332;
    String ubicacion = "Ciudad de México";

    municipioService.guardarUbicacion(latitud, longitud, ubicacion);

    verify(mapaDao, times(1)).guardarUbicacion(latitud, longitud, ubicacion);
}

```

Figura 34 Prueba de guardado de ubicación

La **figura 34** presenta una prueba unitaria que verifica el método *guardarUbicacion*, cuyo objetivo es asegurarse de que funcione correctamente al guardar una ubicación en un mapa. Este método recibe tres parámetros: *latitud*, *longitud* y *una descripción* de la ubicación. Para la prueba, se establecen valores de ejemplo para la latitud (19.432608), la longitud (-99.133209) y la descripción ("Ciudad de México"). Luego, se llama al método *guardarUbicacion* con estos valores. Con ello, se verifica que el método del *MapaDao* sea llamado exactamente una vez con los mismos parámetros de latitud, longitud y descripción.

12. Conclusiones

El desarrollo del módulo de registro del distribuidor ha sido un paso crucial en la mejora de los flujos de comunicación dentro de Pintumex. En el pasado, el proceso de registro y evaluación de nuevas empresas era un proceso manual que implicaba múltiples visitas a las instalaciones y cuestionarios presenciales. Esta metodología no solo era consumidora de tiempo, sino que también generaba retrasos en la plataforma de Pintumex.

La implementación del módulo de registro del distribuidor ha transformado radicalmente este proceso. Ahora, los solicitantes pueden completar los formularios de registro de manera digital, eliminando la necesidad de visitas físicas y agilizando la recopilación de datos. Además, el módulo facilita la comunicación entre el administrador y los distribuidores prospectos, al centralizar toda la información relevante en una plataforma accesible para ambas partes.

Es importante resaltar el impacto positivo que este desarrollo ha tenido en la implementación del proyecto en su conjunto. El módulo de registro del distribuidor proporciona las pautas necesarias para guiar eficazmente todas las etapas del proyecto, desde la solicitud inicial hasta la aprobación final del distribuidor. Gracias a esta herramienta, se ha establecido un proceso estandarizado y eficiente que garantiza la calidad y coherencia en la evaluación de los distribuidores potenciales.

Desde un punto de vista personal, este proyecto me ha permitido aplicar mis conocimientos en la gestión de soluciones informáticas. A través de la implementación de las diversas fases del módulo de registro del distribuidor, he obtenido una comprensión más profunda de los pasos necesarios para llevar a cabo con éxito un proyecto de esta magnitud.

Aunque previamente había participado en fases individuales de proyectos similares, este ejercicio me ha enseñado la importancia de considerar todas las etapas del proceso en su conjunto para asegurar el éxito y la viabilidad de la solución propuesta.

El módulo de registro del distribuidor ha demostrado ser una herramienta invaluable para agilizar los procesos de comunicación y registro dentro de Pintumex. su implementación ha mejorado significativamente la eficiencia operativa de la empresa y ha establecido las bases para una gestión más efectiva de proyectos futuros.

Además, este proyecto ha representado una oportunidad para poner en práctica y consolidar mis habilidades en la gestión de soluciones informáticas.

13. *Desempeño laboral*

13.1 Experiencia laboral

El sistema que se desarrolló me permitió implementar conceptos, prácticas y criterios que ya habían sido vistos durante todo mi proceso de la carrera universitaria, pero ahora dentro de una empresa con un proyecto grande.

Tras recibir la capacitación adecuada, pude adaptarme a cada una de las actividades que me otorgaron, aumentar mi conocimiento sobre algunas herramientas de tecnología, desarrollar habilidades en un modelo de desarrollo de software y desarrollar un sistema desde el back-end como fron-end, desde cero.

13.2 Competencias desarrolladas

- Colaborar eficazmente con equipos multidisciplinarios, incluyendo desarrolladores, diseñadores y testers, para lograr objetivos comunes.
- Gestionar proyectos utilizando metodologías ágiles como Scrum para mejorar la eficiencia y la adaptabilidad del equipo.
- Transmitir ideas y conceptos técnicos de manera clara tanto a compañeros técnicos como a no técnicos.
- Ajustarse rápidamente a nuevas herramientas y tecnologías para mantenerse al día con las demandas del proyecto.
- Buscar y proponer nuevas tecnologías y enfoques que puedan mejorar la eficiencia y la calidad del proyecto.

- Resolver diferencias de opinión y encontrar soluciones que satisfagan a todas las partes involucradas en el proyecto.
- Priorizar tareas y gestionar el tiempo de manera efectiva para cumplir con los plazos del proyecto.
- Revisar meticulosamente el código y las interfaces de usuario para asegurar que no haya errores y que todo funcione correctamente.

Referencias consultadas

1. asana. (5 de Enero de 2024). *Figma + Asana*. Obtenido de <https://asana.com/es/apps/figma>
2. Busquets, C. (8 de septiembre de 2023). *Introducción a Figma: primeros pasos*. Obtenido de <https://www.uifrommars.com/figma-primeros-pasos/>
3. *Compara Software*. (23 de 09 de 2023). Obtenido de Lucidchart: <https://blog.comparasoftware.com/lucidchart/>
4. FANDOM. (10 de 05 de 2023). *Diagrama de casos de uso*. Obtenido de https://la-wikitsm41.fandom.com/es/wiki/Diagrama_de_Casos_de_Uso
5. GEEKNETIC. (28 de 01 de 2024). *Todo sobre Windows 10*. Obtenido de <https://www.geeknetic.es/Guia/1668/Windows-10-Todo-lo-que-necesitas-saber.html>
6. IBM. (17 de 2 de 2023). *¿Qué es Java?* Obtenido de <https://www.ibm.com/mx-es/topics/java>
7. IBM. (2 de Febrero de 2024). *WebSphere Application Server Network*. Obtenido de <https://www.ibm.com/docs/es/was-nd/8.5.5?topic=files-javaserver-faces>
8. miro. (23 de 02 de 2024). *Diagrama de secuencia UML*. Obtenido de <https://miro.com/es/diagrama/que-es-diagrama-secuencia-uml/>
9. Oracle. (12 de 08 de 2023). *La forma más inteligente y rápida de programar*. Obtenido de <https://www.oracle.com/mx/tools/technologies/netbeans-ide.html>

10. Partner, G. M. (25 de Enero de 2024). *Navicat Premium*. Obtenido de <https://www.navicat.com/es/products/navicat-premium#:~:text=Navicat%20Premium%20es%20una%20herramienta,SQLite%20desde%20una%20sola%20aplicaci%C3%B3n>.
11. Pérez, F. (29 de abril de 2023). *TortoiseSVN*. Obtenido de https://tortoisesvn.net/docs/release/TortoiseSVN_es/tsvn-preface.html
12. Pérez, J. H. (16 de marzo de 2022). *Planner Office 365*. Obtenido de <https://www.uaa.mx/alumnos/apoyo/pdf/PLANNER.pdf>
13. Pre, I. (2022 de Marzo de 18). *DEVOPS SERVICIOS GESTIONADOS*. Obtenido de Metodología SCRUM: qué es y cómo implementarlo: <https://ilimit.com/blog/metodologia-scrum/#:~:text=La%20metodolog%C3%ADa%20SCRUM%20consiste%20en,de%20esta%20filosof%C3%ADa%20de%20trabajo>.
14. PrimeFaces. (20 de Enero de 2024). *Código Xules*. Obtenido de <https://codigoxules.org/primefaces/>
15. Services, G. C. (18 de septiembre de 2021). *Soporte GlassFish*. Obtenido de <https://genos.es/glassfish/>
16. Soft, R. (23 de marzo de 2023). *SQL SERVER MANAGEMENT STUDIO: QUÉ ES, VENTAJAS Y CÓMO DESCARGARLO*. Obtenido de <https://blog.revolutionsoft.net/sql-server-management-studio/>
17. Juárez, B. H. (18 de 07 de 2023). *CONSULTING Y LEGAL SERVICES*. Obtenido de <https://www.phconsultingls.com/elaboracion-de-una-minuta/>