



EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA



TECNOLÓGICO
NACIONAL DE MÉXICO®

Instituto Tecnológico de Apizaco

TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE APIZACO

**DESARROLLO E INTEGRACIÓN DEL MÓDULO DE ÓRDENES
DE PAGO AL SISTEMA INTEGRAL DE COBROS MUNICIPALES**

TITULACIÓN INTEGRAL INFORME DE RESIDENCIA PROFESIONAL

**PARA OBTENER EL TÍTULO PROFESIONAL DE:
INGENIERO EN TECNOLOGÍAS DE LA INFORMACIÓN Y
COMUNICACIONES**

**PRESENTA:
ADAIR GUSTAVO MONTIEL HERNANDEZ**

**NOMBRE DEL ASESOR:
M.C. JUAN RAMOS RAMOS**



APIZACO, TLAX., FEBRERO 2024



EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA



TECNOLÓGICO
NACIONAL DE MÉXICO

Instituto Tecnológico de Apizaco
Sistemas y Computación

Apizaco, Tlaxcala, **19/septiembre/2023**

OFICIO No: SC/OPV/193/23

ASUNTO: Liberación de proyecto
para titulación integral

MARTÍN ROJAS RAMÍREZ
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:	ADAIR GUSTAVO MONTIEL HERNÁNDEZ
CARRERA:	INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIONES
NO. DE CONTROL:	18370981
NOMBRE DEL PROYECTO:	DESARROLLO E INTEGRACIÓN DEL MÓDULO DE ÓRDENES DE PAGO AL SISTEMA INTEGRAL DE COBROS MUNICIPALES.
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®



ALAN AUGUSTO CALLEGOS CUELLAR
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
AAGC/mrol*

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

JUAN RAMOS RAMOS
ASESOR INTERNO



Carretera Apizaco-Tzompantepec, esquina con Av. Instituto Tecnológico S/N,
Conurbado Apizaco-Tzompantepec, Tlaxcala, Mex. C.P. 90491 Tel. (241) 4172010 Ext. 115,
e-mail: sistemas@apizaco.tecnm.mx technm.mx | apizaco.tecnm.mx



2022 Flores
Año de Magón
Rico de Magón



Apizaco, Tlax., **14/diciembre/2023**

No. OFICIO: DEP-CT/878/2023

ASUNTO: AUTORIZACIÓN DE IMPRESIÓN

ADAIR GUSTAVO MONTIEL HERNANDEZ
EGRESADO DE LA CARRERA DE
INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIONES
CON NÚMERO DE CONTROL: 18370981
P R E S E N T E

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 E INTEGRACIÓN DEL MÓDULO DE ÓRDENES DE PAGO AL SISTEMA INTEGRAL DE COBROS MUNICIPALES"**, 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 IMPRESION

Debiendo entregar **3** empastados del mismo, apropiadamente encuadernados, a la brevedad, para presentar su acto de recepción profesional.

Sin otro particular por el momento, le envió un cordial saludo.

ATENTAMENTE

*Excelencia en Educación Tecnológica®
Pensar para Servir, Servir para Triunfar®*


MARTÍN ROJAS RAMÍREZ
JEFE DE LA DIVISIÓN DE ESTUDIOS PROFESIONALES

c.p. archivo



Agradecimientos

Con mucho cariño, quiero agradecer a mi padre Lazaro que siempre me ha brindado su apoyo desde que inicie mi carrera para poder continuar con mis estudios que confió en mí cuando tenía problemas ya que sin su apoyo no estaría en este momento terminado la carrera universitaria. A mi madre Margarita que aún se encuentra en otro lugar mejor siempre me apoyo con el proceso de mis estudios, me brinda su apoyo y atención. A mis hermanos Edgar y William que estuvieron conmigo cuando me la pasaba en la noche haciendo tarea y a veces ayudándome en lo que podían. A Jessica que a lo largo de este proceso me brinda su apoyo incondicional para lograr esta meta importante en mi vida. A mis maestros por su guía y orientación. Su paciencia, su sabiduría y sus consejos me han ayudado a crecer como profesional y como persona ya que sin ellos no hubiera sido posible terminar la carrera especialmente al maestro Juan Ramos que me apoyo con este proyecto por sus buenas enseñanzas. A mis amigos que me ayudaron cuando algún tema no entendía, su comprensión y apoyo.

Resumen

A medida que el tiempo ha pasado el desarrollo web ha ido ganando ventajas ya que los procesos se han ido automatizando y este proyecto se enfoca en la optimización del mismo, implementación de soluciones tecnológicas se erige como una prioridad para mejorar la recaudación y el control de ingresos. Este proyecto de residencias se centra en el enriquecimiento del sistema integral de cobros municipales a través del diseño e integración de un módulo de órdenes de pago, empleando el framework de desarrollo web Django.

Principalmente va a permitir a los contribuyentes efectuar sus pagos de manera más ágil y segura. El marco de desarrollo de Django se presenta como una herramienta ideal ya que presenta estructura, seguridad y flexibilidad en la creación y despliegue de aplicaciones robustas.

En este proyecto el proceso de desarrollo va desde una perspectiva sistemática es decir estableciendo los objetivos claros en términos de funcionalidad, usabilidad y seguridad.

Índice

Introducción.....	1
Descripción de la empresa	2
Problemas a resolver	3
Objetivos.....	3
Justificación	4
Delimitación.....	5
Marco teórico	6
Sistemas de cobro municipales	6
Modernización de la tecnología en la administración	6
Metodología de desarrollo ágil.....	8
Scrum	9
Programación extrema	10
Python.....	12
Django.....	12
Modelo vista plantilla	15
ORM de Django.....	16
PostgreSQL.....	16
Experiencia del usuario y Accesibilidad	17
Integración del sistema	19
Generación de informes	19
Caso de estudio	20
Desarrollo	21
Requerimientos funcionales.....	21
Requerimientos no funcionales.....	22
Diagrama de flujo	23
Casos de uso	24
Casos de abuso	25
Historias de usuario	25
Diagrama de base de datos	28

Maquetación del sistema	29
Proyecto en Django	36
Resultados	49
Plan de Pruebas	61
Evaluación de la usabilidad.....	69
Conclusiones.....	75
Competencias desarrolladas	76
Fuentes de información.....	77
Anexos	78
Carta de Aceptación	78
Carta de Liberación.....	79
Carta de No inconveniencia.....	80

Ilustraciones

Figura 1. Diagrama de Procesos Scrum	9
Figura 2. Modelo Vista Témplate Django	15
Figura 3. ORM de Django	16
Figura 4. Diagrama de Flujo	23
Figura 5. Diagrama caso de uso	24
Figura 6. Diagrama de base de datos	28
Figura 7. Página de inicio de sesión	29
Figura 8. Ventana de opciones	30
Figura 9. Registro de áreas	31
Figura 10. Ventana de Registros	32
Figura 11. Ventana de reportes ordenes	33
Figura 12. Ventana de gestión de conceptos	34
Figura 13. Ventana de listado de conceptos	35
Figura 14. Aplicación instalada em Django	37
Figura 15. Configuración de base de datos	37
Figura 16. Estructura de la aplicación	38
Figura 17. Vista de sección órdenes de pago	38
Figura 18. Vistas crud para áreas	39
Figura 19. Vista de listar y crear orden de pago	39
Figura 20. Vista de pdf de ticket	40
Figura 21. Vistas crud para concepto	40
Figura 22. Vista para lista de reportes	41
Figura 23. Vista de listar y actualizar estatus	41
Figura 24. Vista crear pdf de comprobante	41
Figura 25. Modelos para base de datos	42
Figura 26. Modelos en el administrador de Django	42
Figura 27. Formularios de Django	43
Figura 28. URL en la aplicación principal	43
Figura 29. URLs de la aplicación órdenes de pago	44
Figura 30. Plantillas de la aplicación	44
Figura 31. Inserción de CSS y tipo de letra	45
Figura 32. Datatables y plugins en el footer	45
Figura 33. Inserción de iconos	46
Figura 34. Inserción de mensajes de alerta	46
Figura 35. Script de datatables	47
Figura 36. Esqueleto de plantilla	47
Figura 37. Requerimientos del entorno virtual	48

Figura 38. Inicio de sesión	49
Figura 39. Módulos	49
Figura 40. Sección de órdenes de pago	50
Figura 41. Gestión de áreas	50
Figura 42. Crear nueva área.....	51
Figura 43. Modificar área existente.....	51
Figura 44. Eliminar área	52
Figura 45. Listado de órdenes de pago.....	52
Figura 46. Crear nueva orden de pago	53
Figura 47. Ubicación de guardado ticket	53
Figura 48. Ticket generado	54
Figura 49. Reportes.....	54
Figura 50. Listado de conceptos	55
Figura 51. Crear nuevo concepto	55
Figura 52. Modificar concepto actual.....	56
Figura 53. Eliminar un concepto existente	56
Figura 54. Listado de registros en la sección de caja.....	57
Figura 55. Cambio de estatus a una orden de pago	57
Figura 56. Ubicación de guardado comprobante	58
Figura 57. Comprobante generado	58
Figura 58. Menú y cerrado de sesión	59
Figura 59. Inicio de sesión administrador.....	59
Figura 60. Gestión de modelos a través del administrador	60
Figura 61. Cierre de sesión administrador.....	60

Introducción

Actualmente la gestión de pagos del ayuntamiento de Tzompantepec es gestionado de manera manual por cada uno de nos comisionados en su área, esto a lo largo del tiempo genera problemas en cuanto al manejo de datos ya sea que se haga una pérdida de los mismos o una redundancia lo cual hace que el control y seguimiento de los pagos genere demoras.

La eficiencia en la gestión de los cobros y pagos es esencial para garantizar la correcta recaudación de ingresos y el adecuado funcionamiento de los servicios públicos. En este sentido, la implementación de sistemas integrales de cobros municipales se ha convertido en una herramienta fundamental para optimizar los procesos financieros y administrativos, agilizando la recaudación y mejorando la transparencia en el manejo de los recursos.

La integración de un módulo de órdenes de pago representa un paso significativo en la modernización de la infraestructura tecnológica municipal, permitiendo una gestión más fluida de los procesos de recaudación y facilitando el acceso de los contribuyentes a diferentes opciones de pago. Además, este proyecto se enfocará en garantizar la seguridad de las transacciones, la actualización en tiempo real de los registros contables y la generación automatizada de informes y estadísticas relevantes para la toma de decisiones.

Descripción de la empresa

Nombre:

Presidencia Municipal de Tzompantepec

Misión:

Ser un gobierno responsable, transparente, plural, incluyente y cercano a la gente, que impulse el esfuerzo de los habitantes para el mejoramiento de su calidad de vida; que trabaje con racionalidad, honestidad y transparencia; que se conduzca con eficiencia y firmeza; ofreciendo los servicios necesarios; que promueva la participación de los ciudadanos; que planifique el desarrollo sustentable del territorio municipal y gestione recursos para la ejecución de obras necesarias para lograr la prosperidad de Tzompantepec.

Giro:

Sector publico

Domicilio:

Centro, 90490 Tzompantepec.

RFC:

MTT850101BD6

Ciudad:

Tlaxcala

Telefono:

241 415 23 15

Nombre del presidente Municipal:

Ing. Manuel Ramos Montiel

Problemas a resolver

Se han identificado varias deficiencias en la gestión de pagos y estas deficiencias impactan tanto en la eficiencia de los procesos administrativos como en la experiencia de los contribuyentes, lo que compromete la correcta recaudación de ingresos y la transparencia en el manejo de recursos municipales. Por lo tanto, los siguientes problemas a resolver son:

- **Procesos manuales y tiempo de procesamiento:**
El sistema actual requiere de múltiples intervenciones manuales para registrar, verificar y conciliar los pagos realizados por los contribuyentes. Esto genera un proceso lento y propenso a errores, lo que afecta la agilidad en la actualización de registros y en la confirmación de pagos.
- **Falta de opciones de pago y accesibilidad**
Los contribuyentes tienen limitadas opciones para realizar sus pagos, y es que debe ser el personal quien elabora ese formato de pago manual y lo entregue al contribuyente para realizar que vaya a realizar el pago.
- **Dificultad en el seguimiento y generación de informes**
La ausencia de que estos procesos estén automatizados dificulta el que se puedan elaborar informes automatizados y que estén actualizados en tiempo real al momento de que se hace la consulta ya que manualmente tendría que esperarse hasta un día para ver un informe detallado.
- **Experiencia de contribuyente**
Al realizar pagos puede ser frustrante debido a la falta de opciones modernas y la falta de retroalimentación inmediata sobre el estado de los pagos realizados.

Objetivos

Generales:

Realizar el desarrollo e integración de un modelo de órdenes de pago al sistema integral de cobros municipales utilizando el framework Django, con el propósito de mejorar la eficiencia de la gestión de pagos incrementando la transparencia y ofreciendo una experiencia mejorada tanto para los contribuyentes como para los responsables de la administración que hagan uso directamente del sistema.

Específicos:

- Diseñar el módulo de órdenes de pago

Realizar la definición de los requisitos funcionales y no funcionales del módulo, así como la interfaz de usuario, los procesos de verificación y la generación de comprobantes.

- Desarrollar la aplicación con Django
Utilizar el framework Django para crear la aplicación que permita a los contribuyentes recibir en tiempo y forma sus comprobantes de pago
- Integrar el módulo con el sistema integral de cobros
Garantizar la interoperabilidad entre el nuevo mulo y los demás que ya existen en el sistema
- Implementar seguridad y protección de datos
Incorporar medidas de seguridad como como lo es el uso de los mixins que el mismo framework nos brinda
- General informes
Implementar la funcionalidad para generar informes en tiempo real sobre las ordenes generadas
- Realizar pruebas y ajustes
Llevar a cabo pruebas para identificar posibles fallos y asegurarse de que la aplicación funcione de manera óptima y así mismo realizar ajustes y mejora en base a los resultados que se obtengan en las pruebas

Justificación

La implementación del módulo de órdenes de pago en el sistema integral de cobros municipales surge como una respuesta estratégica a las necesidades que enfrenta la administración. Esta iniciativa se justifica en función de varios factores clave que demuestran su relevancia y potencial impacto positivo en la gestión y experiencia de los contribuyentes. Mejorando la eficiencia administrativa reduce la intervención manual, minimizando la posibilidad de errores y acelerando la actualización, transparencia y control ofrecerá a los responsables un mayor control y visibilidad sobre las transacciones en tiempo real. Esto promoverá la transparencia en la gestión de los recursos.

Delimitación

Se encuentra delimitado en varios aspectos clave para garantizar su ejecución eficiente y efectiva en el contexto específico del Municipio de Tzompantepec, Apizaco. El ámbito geográfico se llevará a cabo exclusivamente en el Municipio de Tzompantepec, Apizaco, estará diseñado para ser utilizado por contribuyentes, la funcionalidad se centra en la generación y gestión de órdenes de pago, la implementación se llevará a cabo dentro del presupuesto asignado por el municipio, se utilizará el framework de Django para el desarrollo de todo el módulo.

Marco teórico

El presente proyecto se enmarca en la necesidad de abordar los desafíos a la gestión financiera municipal y la modernización de los sistemas de cobro en el entorno local. Para contextualizar y fundamentar esta iniciativa, es crucial explorar diversas áreas de conocimiento que respalden la importancia de implementar un módulo de órdenes de pago en el sistema integral de cobros municipales utilizando el framework Django.

Sistemas de cobro municipales

Desempeñan un papel fundamental en la administración financiera de los gobiernos locales, ya que permiten la recaudación de los recursos necesarios para el funcionamiento y la prestación de servicios públicos esenciales a la comunidad. Estos sistemas abarcan un conjunto de procesos y mecanismos diseñados para recolectar los ingresos municipales provenientes de diversas fuentes, como impuestos, tasas, contribuciones y derechos, con el objetivo de financiar proyectos y programas que beneficien a los ciudadanos y mejoren la calidad de vida en el entorno local.

Los sistemas de cobro municipales se definen como la infraestructura organizativa y tecnológica utilizada para administrar, supervisar y recaudar los ingresos provenientes de la ciudadanía y otras fuentes, en cumplimiento de las obligaciones tributarias y financieras. La importancia de los sistemas de cobro municipales radica en su papel vital en la sostenibilidad financiera de las administraciones locales. Estos sistemas son esenciales para garantizar la continuidad de los servicios públicos, la inversión en infraestructura y el cumplimiento de los compromisos municipales. Además, una gestión eficiente de los ingresos municipales permite a los gobiernos locales responder a las demandas y necesidades cambiantes de la comunidad, asegurando la equidad en la distribución de la carga tributaria y la eficacia en la asignación de recursos.

Modernización de la tecnología en la administración

Se refiere a la incorporación y uso estratégico de tecnologías de la información y la comunicación (TIC) para mejorar la eficiencia, la transparencia y la calidad de los servicios gubernamentales. En un contexto en constante cambio, la adopción de tecnología es esencial para optimizar los procesos administrativos y brindar servicios más efectivos a los ciudadanos.

Las principales tendencias son:

- Digitalización de servicios
Migran a servicios en línea o en la nube permitiendo acceder de manera remota, lo que agiliza los procesos
- Analítica y Big Data
Para realizar la recopilación y análisis de grandes volúmenes de datos
- Transparencia
Facilita la divulgación y consulta de la información en cualquier momento y a través de cualquier lugar

Las principales ventajas son:

- Eficiencia
Automatizar y digitalizar reduce el tiempo y los recursos dedicado a lo que se realiza manualmente, mejorando la velocidad y precisión de los procesos.
- Transparencia
Permite la trazabilidad y seguimiento en tiempo real
- Reducción de errores
Disminuye la probabilidad de errores en cálculos y registros
- Mejora la experiencia del usuario
Los contribuyentes pueden acceder en línea y obtener información de manera conveniente mejorando su satisfacción

Desafíos principales que se pueden enfrentar

- Costo
La implementación y mantenimiento pueden requerir inversiones significativas
- Cambio cultural
La adopción tecnológica puede encontrarse con resistencia lo que requiere un proceso de adaptación y capacitación
- Seguridad
Las soluciones tecnológicas deben ser seguras contra amenazas
- Accesibilidad
Garantizar que sean accesibles para todos incluyendo aquellos con discapacidades

Metodología de desarrollo ágil

Según Red Hat el enfoque ágil para el desarrollo de software busca distribuir de forma permanente sistemas de software en funcionamiento diseñados con iteraciones rápidas. Las metodologías ágiles de desarrollo de software buscan proporcionar en poco tiempo pequeñas piezas de software en funcionamiento para aumentar la satisfacción del cliente. Estas metodologías utilizan enfoques flexibles y el trabajo en equipo para ofrecer mejoras constantes. Por lo general, el desarrollo ágil de software implica que pequeños equipos autoorganizados de desarrolladores y representantes empresariales se reúnan regularmente en persona durante el ciclo de vida del desarrollo de software. La metodología ágil favorece un enfoque sencillo de la documentación de software y acepta los cambios que puedan surgir en las diferentes etapas del ciclo de vida, en lugar de resistirse a ellos.

Valores de la metodología

- Las personas y las interacciones antes que los procesos y las herramientas
- El software en funcionamiento antes que la documentación exhaustiva
- La colaboración con el cliente antes que la negociación contractual
- La respuesta ante el cambio antes que el apego a un plan

Los enfoques ágiles de trabajo se crearon para abordar las limitaciones del método en cascada, que se derivaba del sistema de cadena de montaje de 1913 que utilizó Henry Ford como método de fabricación, y que luego se aplicó al desarrollo de software. Desde su origen en 2001, el desarrollo ágil pisa fuerte en el sector de los sistemas de software y la gestión de proyectos, a pesar de sus múltiples variantes.

Manifiesto ágil

Es un documento redactado en 2001 por 17 expertos en programación que supuso un cambio radical en la forma de desarrollar 'software'. Frente a los modelos tradicionales excesivamente rígidos y alejados de las necesidades de los clientes, estos gurús propusieron cuatro valores que inspiran las diferentes metodologías ágiles que han surgido desde entonces.

Valores del manifiesto ágil

- Individuos e interacciones sobre procesos y herramientas.
- 'Software' funcionando sobre documentación exhaustiva.
- Colaboración con el cliente sobre negociación contractual.
- Respuesta ante el cambio sobre seguir un plan.

Scrum

Según Ikujiro Nonaka e Hirotaka Takeuchi creadores de scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos.

Se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

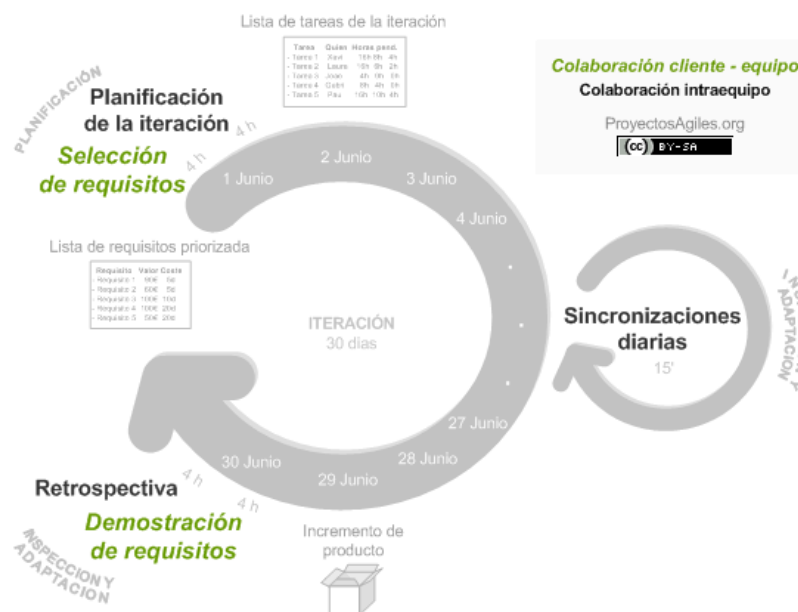


Figura 1. Diagrama de Procesos Scrum

- Planificación de la iteración
 - El primer día de la iteración se realiza la reunión de planificación de la iteración
 - Selección de requisitos

El cliente presenta al equipo la lista de requisitos priorizada del producto o proyecto. El equipo pregunta al cliente las dudas que surgen y selecciona los requisitos más prioritarios que prevé que podrá completar en la iteración, de manera que puedan ser entregados si el cliente lo solicita.

- Planificación de la iteración

El equipo elabora la lista de tareas de la iteración necesarias para desarrollar los requisitos seleccionados. La estimación de esfuerzo se hace de manera conjunta y los miembros del equipo se autoasignan las tareas, se autoorganizan para trabajar incluso en parejas (o grupos mayores) con el fin de compartir conocimiento (creando un equipo más resiliente) o para resolver juntos objetivos especialmente complejos.

- Ejecución de la iteración

Cada día el equipo realiza una reunión de sincronización (15 minutos), normalmente delante de un tablero físico o pizarra (Scrum Taskboard). El equipo inspecciona el trabajo que el resto está realizando (dependencias entre tareas, progreso hacia el objetivo de la iteración, obstáculos que pueden impedir este objetivo) para poder hacer las adaptaciones necesarias que permitan cumplir con la previsión de objetivos a mostrar al final de la iteración.

- Inspección y adaptación

El último día de la iteración se realiza la reunión de revisión de la iteración

- Revisión

El equipo presenta al cliente los requisitos completados en la iteración, en forma de incremento de producto preparado para ser entregado con el mínimo esfuerzo. En función de los resultados mostrados y de los cambios que haya habido en el contexto del proyecto, el cliente puede realizar las adaptaciones necesarias de manera objetiva, ya desde la primera iteración, replanificando el proyecto.

- Retrospectiva

El equipo analiza cómo ha sido su manera de trabajar y cuáles son los problemas que podrían impedirle progresar adecuadamente, mejorando de manera continua su productividad. El Facilitador se encargará de eliminar o escalar los obstáculos identificados que estén más allá del ámbito de acción del equipo.

Programación extrema

Según Alicia Raeburn es una metodología ágil de gestión de proyectos que se centra en la velocidad y la simplicidad con ciclos de desarrollo cortos. Esta metodología se basa en 5

valores, 5 reglas y 12 prácticas de programación. Si bien tiene una estructura rígida, el resultado de estos Sprint altamente centrados y las integraciones continuas buscan dar como resultado un producto de mayor calidad.

Los orígenes de XP se remontan a fines de la década de 1990, cuando Kent Beck la creó para gestionar el desarrollo de un sistema de software de nómina para Chrysler llamado Proyecto C3. El objetivo al implementar la programación extrema era (y sigue siendo) eliminar la resistencia a cambiar el código en un proyecto de desarrollo.

Valores

- | | |
|-----------------|-------------|
| 1. Simplicidad | 4. Valentía |
| 2. Comunicación | 5. Respeto |
| 3. Comentarios | |

Reglas

- | | |
|------------------|-----------------|
| 1. Planificación | 4. Codificación |
| 2. Gestión | 5. Prueba |
| 3. Diseño | |

Las 12 practicas son

- | | |
|----------------------------------|---------------------------------|
| 1. El juego de planificación | 7. Refactorización |
| 2. Prueba de clientes | 8. Propiedad colectiva |
| 3. Pequeñas entregas | 9. Integración continua |
| 4. Diseño simple | 10. Ritmo de trabajo sostenible |
| 5. Programación en parejas | 11. Metáfora |
| 6. Desarrollo guiado por pruebas | 12. Estándares de codificación |

Python

Según aws es un lenguaje de programación ampliamente utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos y el machine learning (ML).

Guido Van Rossum, un programador de computación de los Países Bajos, creó Python. Python comenzó en 1989 en el Centrum Wiskunde & Informatica (CWI), en principio como un proyecto de afición para mantenerse ocupado durante las vacaciones de Navidad. El nombre del lenguaje se inspiró en el programa de televisión de la BBC "Monty Python 's Flying Circus" debido a que Guido Van Rossum era un gran aficionado del programa.

Principales características

- Un lenguaje interpretado

Ejecuta directamente el código, línea por línea. Si existen errores en el código del programa, su ejecución se detiene.

- Un lenguaje fácil de utilizar

Utiliza palabras similares a las del inglés

- Un lenguaje tipeado dinámicamente

Los programadores no tienen que anunciar tipos de variables cuando escriben código porque Python los determina en el tiempo de ejecución.

- Un lenguaje de alto nivel

Es más cercano a los idiomas humanos que otros lenguajes de programación. Por lo tanto, los programadores no deben preocuparse sobre sus funcionalidades subyacentes, como la arquitectura y la administración de la memoria.

- Un lenguaje orientado a los objetos

Considera todo como un objeto, pero también admite otros tipos de programación, como la programación estructurada y la funcional.

Django

Según documentación de Django es un framework web de alto nivel que permite el desarrollo rápido de sitios web seguros y mantenibles. Desarrollado por programadores experimentados, Django se encarga de gran parte de las complicaciones del desarrollo web, por lo que puedes concentrarte en escribir tu aplicación sin necesidad de reinventar la

rueda. Es gratuito y de código abierto, tiene una comunidad próspera y activa, una gran documentación y muchas opciones de soporte gratuito y de pago.

Fue desarrollado inicialmente entre 2003 y 2005 por un equipo que era responsable de crear y mantener sitios web de periódicos. Después de crear varios sitios, el equipo empezó a tener en cuenta y reutilizar muchos códigos y patrones de diseño comunes. Este código común se convirtió en un framework web genérico, que fue de código abierto, conocido como proyecto "Django" en julio de 2005.

Principales características

- **Velocidad de desarrollo**
El marco Django está bien organizado y es fácil de instalar y aprender, por lo que puede comenzar en cuestión de horas. Los diseñadores de Django crearon el marco para implementar rápidamente cualquier arquitectura web en el código. Permite el desarrollo rápido y el diseño limpio y pragmático.
- **Rentabilidad**
Es un proyecto de Python gratuito y de código abierto con una comunidad activa que revisa y mantiene el software. Una organización sin fines de lucro llamada Django Software Foundation promueve y respalda el uso y el mantenimiento de Django. Organiza encuentros, reuniones y eventos comunitarios regulares que animan a otros desarrolladores a revisar y contribuir al proyecto Django.

Estructura de Django

- **Modelo**
Los modelos de Django actúan como interfaz entre la base de datos y el código del servidor. Son la única fuente definitiva de información sobre sus datos. Estos modelos de datos contienen los campos y las operaciones esenciales que necesita para interactuar con la base de datos. Los modelos de Django convierten así las tablas de la base de datos en clases u objetos en el código Python. Esto se denomina "mapeo objeto-relacional".
- **Vista**
Las vistas de Django procesan la solicitud utilizando los modelos. Puede escribir una función de vista para cada tipo de solicitud que los visitantes de su sitio web pueden hacer a este. Una función de vista puede tomar la solicitud como entrada y devolver

una respuesta, que puede ser un código de error, una imagen, un archivo o cualquier tipo de dato.

Django tiene una función de mapeador o despachador de URL, que asigna sus funciones de vista a sus URL.

- **Plantilla**

Las plantillas de Django administran la presentación de la página web en el navegador. Dado que la mayoría de las páginas web están en el lenguaje de marcado de hipertexto (HTML), puede escribir el código de la plantilla de Django en un estilo similar a HTML. Un archivo de plantilla contiene ciertos componentes:

- Las partes estáticas de la salida HTML final, como imágenes, botones y encabezados.
- Sintaxis especial que describe cómo insertar contenido o datos dinámicos, que cambia con cada solicitud.

Módulos extras en Django

- **Formularios**

Django gestiona el procesamiento de formularios de tres maneras:

1. Creación de formularios mediante la preparación y reestructuración de los datos para su visualización
2. Validación de formularios mediante la comprobación de formularios HTML en el lado del cliente
3. Procesamiento de formularios mediante la recepción de los datos enviados

- **Autenticación de usuarios**

Django puede gestionar la autenticación para varios usos.

- Cuentas de usuario
- Permisos e indicadores de sí o no que habilite a los usuarios realizar ciertas tareas del sitio web
- Grupos de cuentas de usuario múltiples con permisos similares
- Sesiones de usuario basadas en cookies

- **Administraciones del sitio**

El sitio de administración de Django facilita la provisión de una página de administración para su sitio. Los administradores del sitio pueden usar la página para crear, editar o ver los modelos de datos del sitio.

Modelo vista plantilla

En Django, el controlador sigue estando presente, nada más que de una manera intrínseca, ya que todo el framework Django es el controlador.

- **Modelo:**
Maneja todo lo relacionado con la información, esto incluye como acceder a esta, la validación, relación entre los datos y su comportamiento.
- **Vista:**
Es un enlace entre el modelo y el template. Decide qué información será mostrada y por cual template.
- **Téplate**
Decide como será mostrada la información.

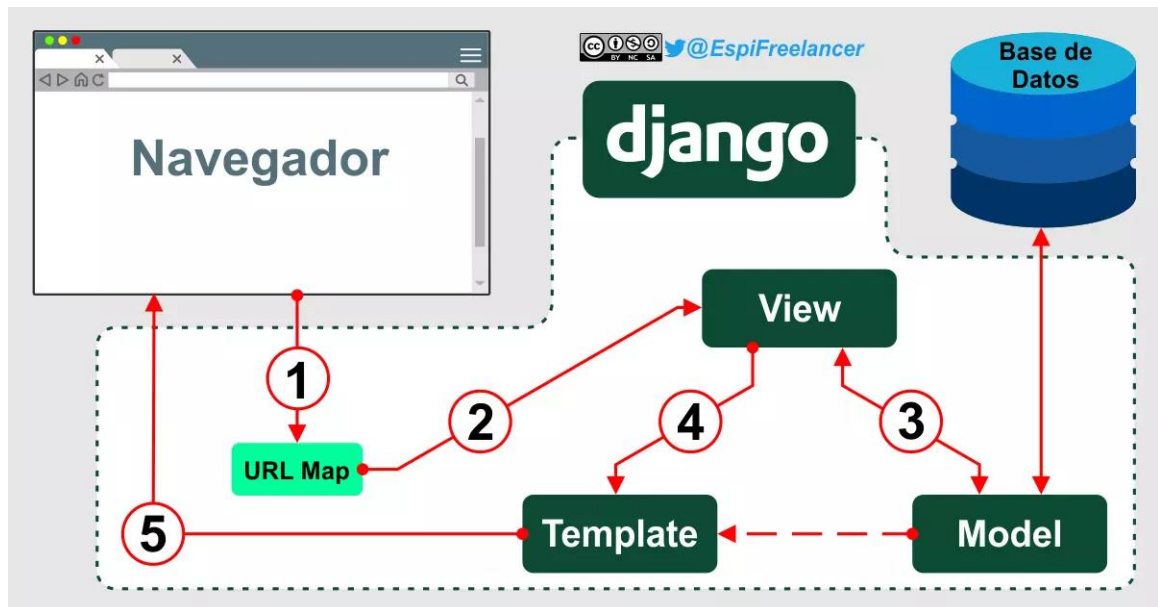


Figura 2. Modelo Vista Téplate Django

Funcionamiento (Explicación de figura 2)

Al momento de hacer clic en un enlace o escribir una dirección (1) a lo primero que se accede es al mapa de URLs (también conocido como URL map o URL conf), en este archivo cada ruta está asociada con una view (2), si se necesita algún dato se solicitara este a model (3) el cual a su vez generara la consulta a la base de datos, cuando los datos han sido traídos estos son enviados al template (4) que contiene la lógica de presentación para estos. Luego de "pintar" la página esta se envía al navegador que hizo la solicitud (5).

ORM de Django

De acuerdo a un curso el ORM es una implementación del concepto de mapeo de objeto relacional (ORM).

Una de las características más poderosas de Django es su Mapeador Relacional de Objetos (ORM), que le permite interactuar con su base de datos, como lo haría con instrucciones SQL (Structured Query Language). De hecho, el ORM de Django es sólo una forma de crear SQL para consultar y manipular su base de datos y obtener resultados.

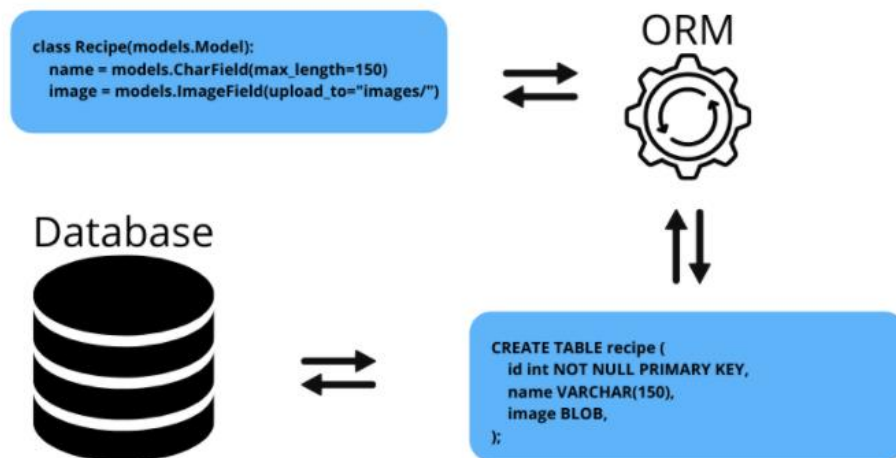


Figura 3. ORM de Django

PostgreSQL

Según la documentación de IBM es una base de datos de código abierto que tiene una sólida reputación por su fiabilidad, flexibilidad y soporte de estándares técnicos abiertos. A diferencia de otros RDMBS (sistemas de gestión de bases de datos relacionales), PostgreSQL soporta tipos de datos relacionales y no relacionales. Esto la convierte en una de las bases de datos relacionales más compatibles, estables y maduras disponibles actualmente.

Importancia de utilizar esta base de datos

PostgreSQL es expandible y versátil, por lo que puede soportar rápidamente una variedad de casos de uso especializados con un poderoso ecosistema de extensión, que abarca desde tipos de datos de series de tiempo hasta análisis geoespaciales.

Beneficios

- Rendimiento y escalabilidad
En sistemas de bases de datos más grandes donde la autenticación de datos y las velocidades de lectura/escritura son esenciales, PostgreSQL es difícil de superar. PostgreSQL soporta diferentes optimizaciones de rendimiento que normalmente solo se encuentran en la tecnología de base de datos patentada, como el soporte geoespacial y la concurrencia sin restricciones.
- Soporte de concurrencia
Cuando varios usuarios acceden a los datos al mismo tiempo, los sistemas de bases de datos tradicionales normalmente bloquean el acceso a los registros para evitar conflictos de lectura/escritura. PostgreSQL gestiona la concurrencia de manera eficiente mediante el uso de MVCC (Control de concurrencia de múltiples variantes).
- Soporte de lenguaje profundo
PostgreSQL es una de las bases de datos más flexibles para desarrolladores debido a su compatibilidad y soporte de múltiples lenguajes de programación. Los lenguajes de codificación populares como Python, JavaScript, C/C ++, Ruby y otros ofrecen un soporte maduro para PostgreSQL, lo que permite a los desarrolladores realizar tareas de bases de datos en el lenguaje que dominan sin generar conflictos en el sistema.
- 100% de código abierto
La implementación de tecnología de gestión de bases de datos de código abierto ofrece beneficios únicos a las empresas, incluidos mejores costos, mayor flexibilidad e innovación que no siempre está disponible con las soluciones de bases de datos patentadas.

Experiencia del usuario y Accesibilidad

La experiencia del usuario (UX) y la accesibilidad son aspectos fundamentales en el diseño y desarrollo de aplicaciones web. Brindar una experiencia satisfactoria a los contribuyentes y garantizar que la aplicación sea accesible para todos los usuarios son objetivos esenciales para lograr la adopción exitosa del módulo de órdenes de pago en el sistema integral de cobros municipales.

Según Juan Andrés Corrales engloba todas aquellas interacciones del usuario con una marca. Por supuesto, a raíz de la transformación digital, esta experiencia se orienta principalmente a páginas web, aplicaciones móviles, herramientas y otros recursos similares.

Importancia de brindar una experiencia satisfactoria

- **Fidelización del Usuario**
Una experiencia positiva aumenta la probabilidad de que los usuarios permanezcan en el sistema y este lo adopten sin problema
- **Confianza**
Una interfaz intuitiva y fluida transmite profesionalismo y confiabilidad, generando confianza en los usuarios al utilizar el sistema
- **Eficiencia**
Un diseño bien pensado y funcionalidades claras agilizan el de generación de órdenes, ahorrando tiempo a los contribuyentes y mejorando la eficiencia del sistema.
- **Reducción de errores**
Una interfaz intuitiva y amigable minimiza la probabilidad de errores

Consideración de Accesibilidad y usabilidad

- **Diseño centrado en el usuario**
El diseño debe considerar las necesidades y expectativas de los diferentes usuarios
- **Usabilidad**
La aplicación debe ser fácil de entender y usar, evitando complicaciones innecesarias en la generación de órdenes de pago.
- **Claridad y sencillez**
La información debe presentarse de manera clara y concisa, utilizando un lenguaje simple y directo.
- **Diseño responsivo**
La aplicación debe adaptarse a diferentes dispositivos y tamaños de pantalla, asegurando una experiencia uniforme en computadoras de escritorio, tabletas y teléfonos móviles
- **Pruebas con usuarios**
Realizar pruebas de usabilidad con usuarios reales ayuda a identificar áreas de mejora y a garantizar que la aplicación cumpla con sus necesidades.

Integración del sistema

- Interconexión de los componentes
La integración implica la conexión y la comunicación entre diferentes sistemas y aplicaciones, permitiendo que trabajen juntos de manera armoniosa.
- Flujo de datos
La integración implica la transferencia bidireccional de datos y la sincronización de información entre sistemas y bases de datos.
- Consistencia de datos
La integración busca mantener la coherencia y la precisión de los datos a lo largo de los sistemas, evitando duplicaciones y discrepancias.
- Compatibilidad
Los sistemas existentes y las nuevas soluciones pueden estar contruidos con tecnologías y estándares diferentes, lo que puede dificultar la interoperabilidad.
- Seguridad y privacidad
La integración debe garantizar que la seguridad de los datos se mantenga en todo momento y que los protocolos de privacidad se cumplan.
- Planificación detallada
Una planificación meticulosa es esencial para evitar sorpresas y asegurar una integración exitosa.
- Uso de estándares
El uso de estándares de la industria y protocolos de comunicación facilita la integración.
- Monitoreo continuo
Una vez integrados, los sistemas deben ser monitoreados continuamente para detectar problemas.

Generación de informes

Desempeña un papel crucial en la toma de decisiones informadas en la administración municipal. Proporciona una visión clara y objetiva de la situación, operativa y de rendimiento, lo que permite a los responsables tomar medidas basadas en datos con el objetivo de mejorar la eficiencia y la calidad de los servicios proporcionados.

Importancia de generar informes

- Base de datos
La toma de decisiones informada se basa en datos concretos en lugar de suposiciones o intuiciones, lo que lleva a resultados más precisos y efectivos.
- Transparencia
La generación de informes brinda transparencia a los ciudadanos y las partes interesadas, ya que pueden acceder a información objetiva sobre las operaciones municipales.
- Rendimiento
Los informes permiten evaluar el rendimiento de los servicios, identificar áreas de mejora y tomar medidas para optimizar los procesos.

Caso de estudio

El Municipio de Tzompantepec ubicado en Apizaco, en el estado de Tlaxcala, México. Con una población diversa y activa, el municipio enfrenta desafíos comunes a muchas áreas urbanas, como la prestación de servicios públicos, la infraestructura y la recaudación de ingresos para financiar estos servicios. Tzompantepec se caracteriza por su rica historia, cultura y potencial económico, lo que subraya la importancia de una gestión financiera eficiente y efectiva para el bienestar de sus habitantes.

Situación Actual y oportunidades de mejora

- Procesos manuales
Los procesos manuales pueden ser propensos a errores y demorados, lo que afecta la eficiencia y la precisión en la gestión financiera.
- Transparencia y accesibilidad
La necesidad de mejorar la transparencia y facilitar la accesibilidad a los servicios municipales a través de soluciones tecnológicas.
- Experiencia de usuario
Brindar una experiencia satisfactoria a los contribuyentes y acceder a información relacionada con sus obligaciones.
- Información centralizada
La falta de una plataforma centralizada para gestionar y administrar las órdenes de pago.

Desarrollo

Requerimientos funcionales

- Registros de las áreas del municipio
 - El sistema debe permitir al personal registrar nuevas áreas
 - Se deben proporcionar campos para el nombre y la descripción de cada una
- Registro de órdenes de pago
 - El sistema debe permitir registrar órdenes de pago
 - Se deben proporcionar campos para el folio que va hacer único, el nombre del solicitante, la descripción, el área a la que va a pertenecer el ticket, el concepto por el cual se está emitiendo, y la fecha en la que se está emitiendo
- Consulta de registros
 - El sistema debe permitir visualizar todos los reportes que se han generado
 - Se deben proporcionar opciones como lo son copiar, exportar a csv, Excel, pdf o imprimir y buscar
- Registro de conceptos
 - El sistema debe permitir registrar conceptos los cuales se puede hacer el cobro
 - Se debe proporcionar campos para el nombre del concepto, el monto y el área al que pertenece
- Registro de caja
 - El sistema debe permitir visualizar los tickets que se han generado
 - Se debe proporcionar campos para marcar si el ticket ha sido pagado o aun esta por pagar además de generar el ticket
- Acceso en línea
 - El personal debe tener acceso en línea para ver y gestionar los pendientes
- Accesibilidad y usabilidad
 - La interfaz del sistema debe ser accesible y fácil de usar, brindando una experiencia satisfactoria para personal con diferentes niveles de habilidades tecnológicas
- Soporte y asistencia
 - Debe proporcionar una canal de soporte para el personal en caso de problemas técnicos o dudas relacionadas con el sistema y los pagos

Requerimientos no funcionales

- Rendimiento
 - El sistema debe ser capaz de manejar un volumen razonable de órdenes de pago y transacciones simultáneas sin degradación significativa del rendimiento
- Tiempo de respuesta
 - El tiempo de respuesta para la generación de órdenes de pago, el acceso a la plataforma y la realización de pagos en línea debe ser rápido y satisfactorio para los usuarios
- Seguridad de datos
 - El sistema debe cumplir con estándares de seguridad y proteger los datos financieros y personales de los contribuyentes, utilizando encriptación y prácticas de seguridad robustas
- Disponibilidad
 - El sistema debe estar disponible y accesible para el personal la mayor parte del tiempo, con un tiempo de inactividad planificado mínimo para mantenimiento
- Escalabilidad
 - El sistema debe ser escalable y capaz de manejar un aumento en la demanda de usuarios y transacciones a medida que crece el número de contribuyentes
- Compatibilidad
 - La aplicación debe ser compatible con diferentes navegadores web, sistemas operativos y dispositivos, garantizando una experiencia uniforme para todos los usuarios
- Usabilidad
 - La interfaz debe ser intuitiva y fácil de usar, con una curva de aprendizaje mínima para los contribuyentes
- Mantenibilidad
 - El código y la arquitectura deben ser organizados y documentados, facilitando futuras actualizaciones y mejoras
- Respuesta a fallos
 - El sistema debe contar con medidas de contingencia para responder a posibles fallos, minimizando la pérdida de datos y el tiempo de inactividad

Diagrama de flujo

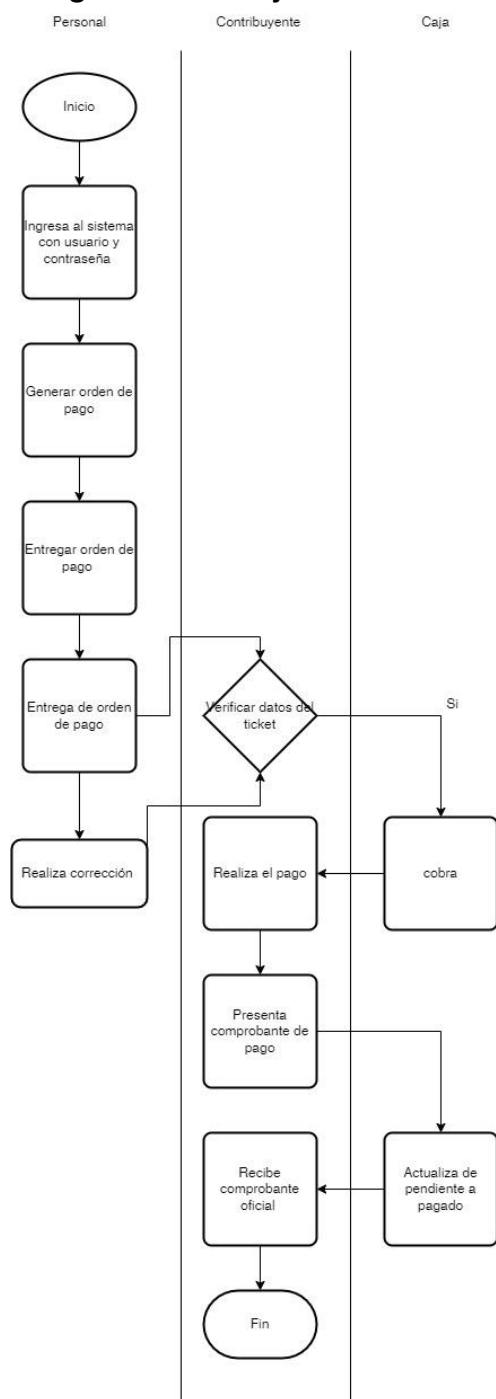


Figura 4. Diagrama de Flujo

Este diagrama muestra el proceso que se realiza desde que el contribuyente llega con el personal a generar una orden de pago, hasta que va a caja a realizar el pago y regresa a validar su pago.

Casos de uso

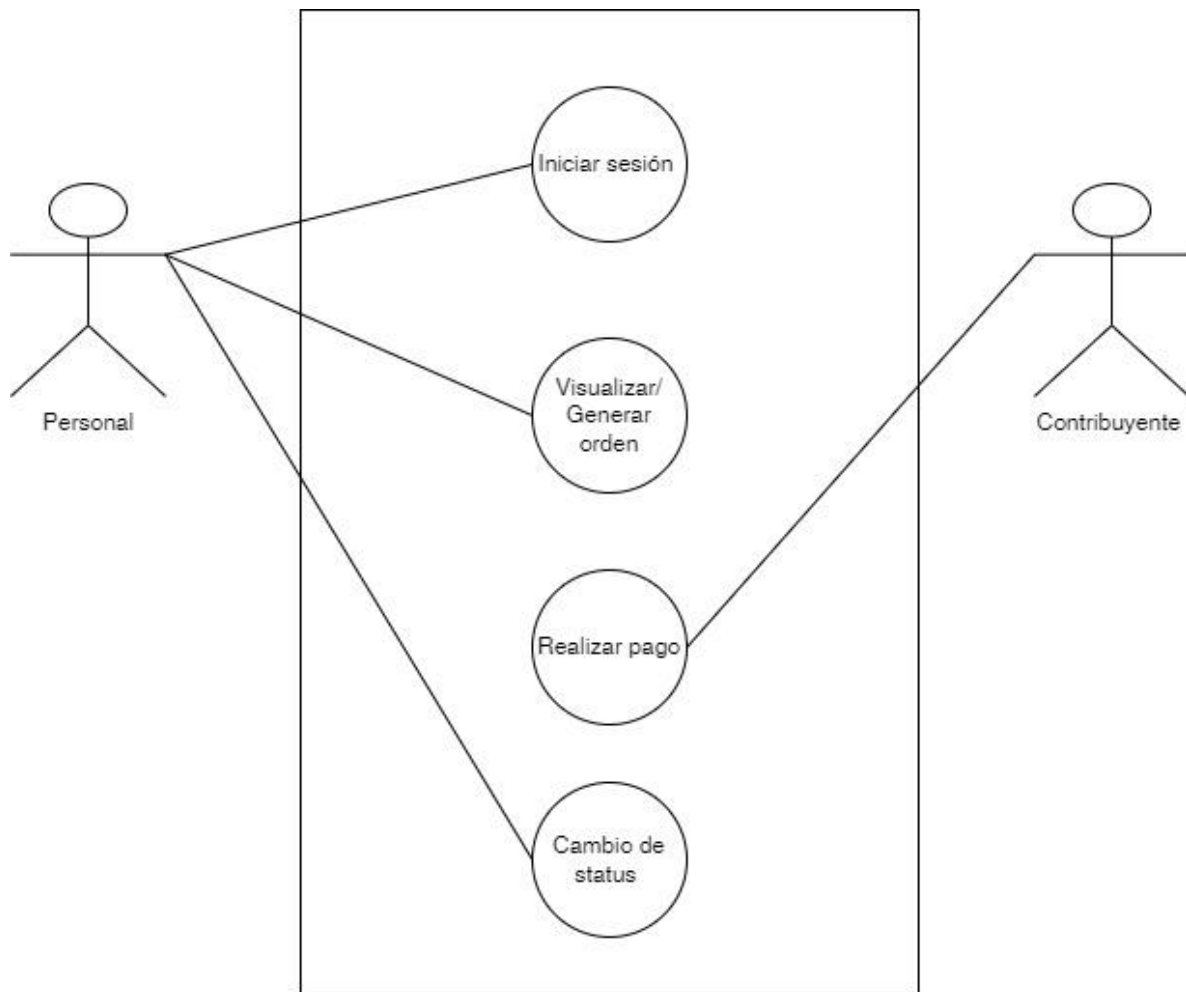


Figura 5. Diagrama caso de uso

En este diagrama hay dos actores el contribuyente y el personal encargado de administrar el sistema, generar órdenes de pago e informes

- Iniciar sesión: el personal inicia con su usuario y contraseña en el sistema para acceder a su cuenta
- Visualizar / Generar ordenes: el personal puede ver una lista de las órdenes de pago pendiente mostrando el detalle con el monto y la fecha y también ve una lista de ordenes que ya están pagadas
- Realizar el pago: el contribuyente acude a caja a realizar el pago
- Cambio de status: el personal recibe el ticket y cambia el status a pagado

Casos de abuso

- ✓ Generación Fraudulenta de órdenes de pago
Un usuario con acceso no autorizado intenta generar órdenes de pago falsas o alteradas para beneficio propio
- ✓ Acceso no autorizado
Un usuario intenta acceder a órdenes de pago de otros contribuyentes utilizando técnicas de hackeo o suplantación de identidad
- ✓ Sobrecarga de pagos
Un usuario intenta sobrecargar el sistema realizando una gran cantidad de pagos en un corto período de tiempo para afectar su funcionamiento
- ✓ Distribución de malware
Un atacante intenta distribuir malware o phishing a través de correos electrónicos falsos relacionados con el sistema de órdenes de pago
- ✓ Inyección de datos
Un usuario intenta introducir código malicioso o scripts en los campos de entrada para comprometer la seguridad o afectar el funcionamiento del sistema
- ✓ Ataques de denegación de servicio
Un atacante intenta inundar el sistema con un alto volumen de solicitudes para sobrecargarlo y hacer que sea inaccesible
- ✓ Aprovechamiento de errores de diseño
Un usuario identifica y explota vulnerabilidades o errores en el diseño del sistema para acceder a funciones o datos no autorizados
- ✓ Suplantación de identidad
Un atacante intenta hacerse pasar por un funcionario municipal o un contribuyente legítimo para obtener acceso a funciones privilegiadas
- ✓ Uso de cuentas robadas
Un atacante utiliza credenciales robadas para acceder al sistema y realizar actividades maliciosas

Historias de usuario

Rol	Personal	Identificador HU1
Característica	Quiero poder iniciar sesión en el sistema utilizando mi nombre de usuario y contraseña para acceder a mi cuenta personalizada	
Razón	Para ver mis órdenes de pago	
Criterio de aceptación	Pantalla Loguin	

Contexto	Se crea la pantalla con logos oficiales y colores del instituto, mostrando el frame de Loguin
Evento	Pantalla de inicio
Resultado	Se visualiza la pantalla de inicio

Rol	Personal	Identificador HU2
Característica	Quiero tener la capacidad de ver una lista completa de las órdenes de pago	
Razón	Para poder identificar cada orden por su concepto, monto y fecha	
Criterio de aceptación	Pantalla de órdenes de pago	
Contexto	Creación de la pantalla donde muestra dos botones de nuevo y regresar, además de una tabla con la información más relevante de las ordenes	
Resultado	Se visualiza la pantalla	

Rol	Personal	Identificador HU3
Característica	Necesito realizar órdenes de pago en el sistema	
Razón	Para cada orden se debe ingresar el concepto, el monto y el área correspondiente para un registro preciso	
Criterio de aceptación	Pantalla de crear una orden de pago	
Contexto	En la pantalla se muestran los campos que se deben de llenar antes de guardar el formulario o cancelar	
Resultado	Se muestra un botón de guardar que retorna al listado de ordenes	

Rol	Personal	Identificador HU4
Característica	Deseo generar informes y estadísticas relacionados con las órdenes de pago	
Razón	Esto me ayudara a supervisar los realizados	
Criterio de aceptación	Pantalla donde todas las órdenes de pago	
Contexto	En la pantalla se muestra una tabla, con diferentes columnas con la información de cada una	
Resultado	Esos reportes se pueden exportar a diferentes tipos de archivo para mejor manejo de información	

Rol	Personal	Identificador HU5
Característica	Quiero tener acceso en línea para ver y gestionar las	

	ordenes de pagó
Razón	Esto permitirá consultarlas y generar reportes en cualquier momento
Criterio de aceptación	Dominio de internet
Contexto	Estará un dominio destinado al sistema 24/7
Resultado	Desde cualquier dispositivo con acceso a internet se puede acceder al dominio

Rol	Sistema	Identificador HU6
Característica	Se debe implementar una validación exhaustiva de los montos ingresados en las órdenes de pago	
Razón	Para garantizar que seas precisos y ayudaran a prevenir intentos de fraude	
Criterio de aceptación	Pantalla dedicada a conceptos	
Contexto	En la pantalla se suben el nombre de los conceptos con su respectivo monto	
Resultado	Para cada orden de pago se muestra una lista del concepto a agregar	

Rol	Sistema	Identificador HU7
Característica	Se debe registrar las actividades de acceso	
Razón	Permitirá monitoreas la seguridad del sistema y detectar posibles amenazas	
Criterio de aceptación	Subdominio del administrador del sistema	
Contexto	El sistema registra actividades de los usuarios	
Resultado	Se registra del usuario fecha y hora en la que accedió al sistema	

Diagrama de base de datos

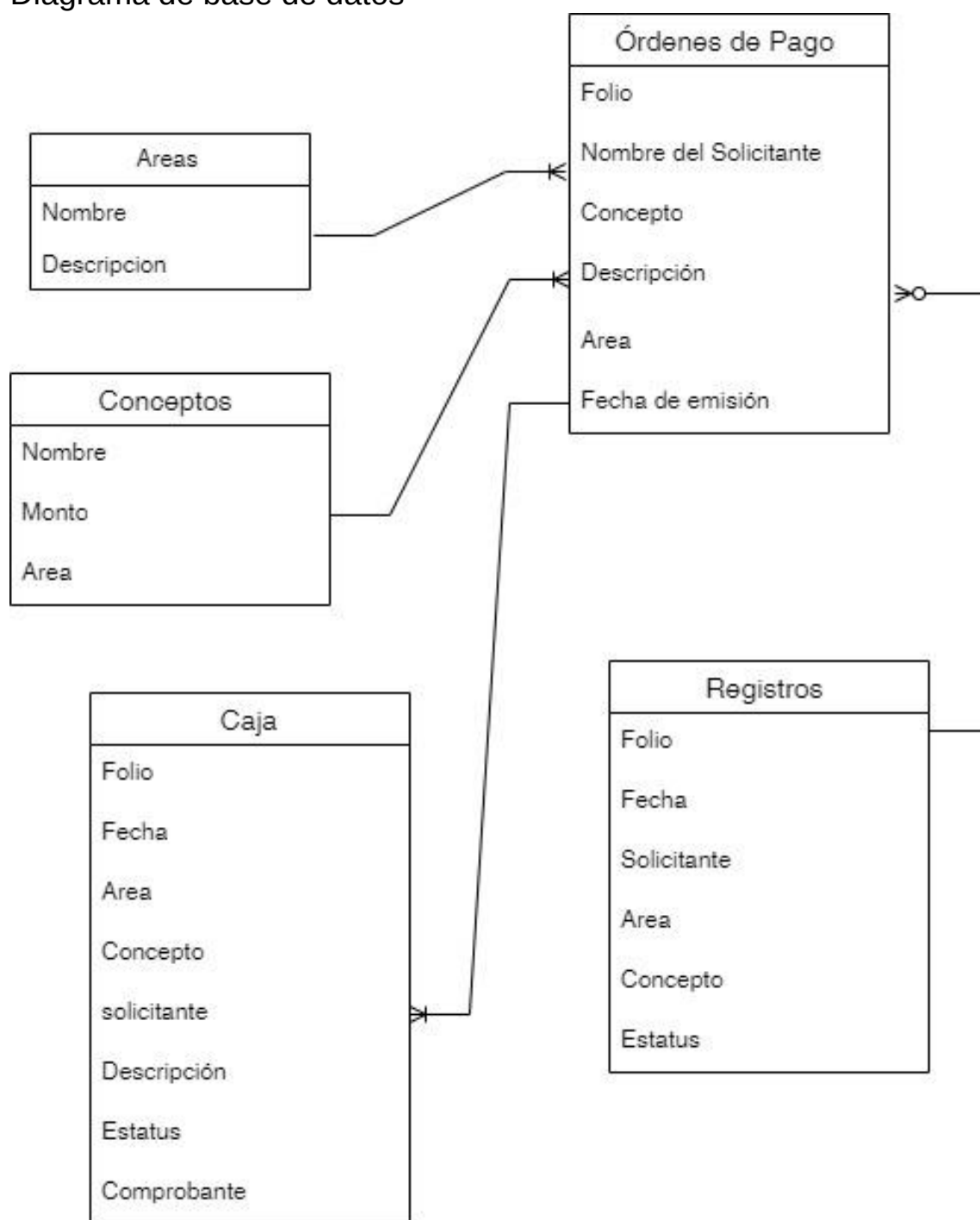


Figura 6. Diagrama de base de datos

Maquetación del sistema

Se muestra maquetación general del sistema desde la pantalla de inicio hasta todos los procesos que se contemplan en el sistema

A Web Page

https://www.tzompantepec.com

TZOMPANTEPEC

Bienvenido

+

Usuario

contraseña

Entrar

The image is a wireframe of a login page for 'TZOMPANTEPEC'. It features a browser window header with navigation icons and a search bar. The main content area has the site name 'TZOMPANTEPEC' in large letters, followed by the word 'Bienvenido' (Welcome). Below this is a circular icon containing a plus sign, representing a user. Underneath the icon are two rectangular input fields labeled 'Usuario' (Username) and 'contraseña' (password). At the bottom of the form is a rectangular button labeled 'Entrar' (Login). The entire page is enclosed in a simple rectangular border.

Figura 7. Página de inicio de sesión

Esta ventana muestra, donde el usuario puede iniciar sesión, en el encabezado se va a encontrar el nombre del municipio, centrado se va a dar la bienvenida, seguido de un icono que representa a un usuario, posteriormente dos rectángulos donde se va a introducir el usuario y contraseña del usuario y para terminar se va a encontrar el botón de entrar.

Si el usuario llega a introducir algún dato erróneo abajo del botón el sistema va a mostrar un mensaje indicando que se ha introducido el usuario o contraseña incorrectamente y que inicie nuevamente.

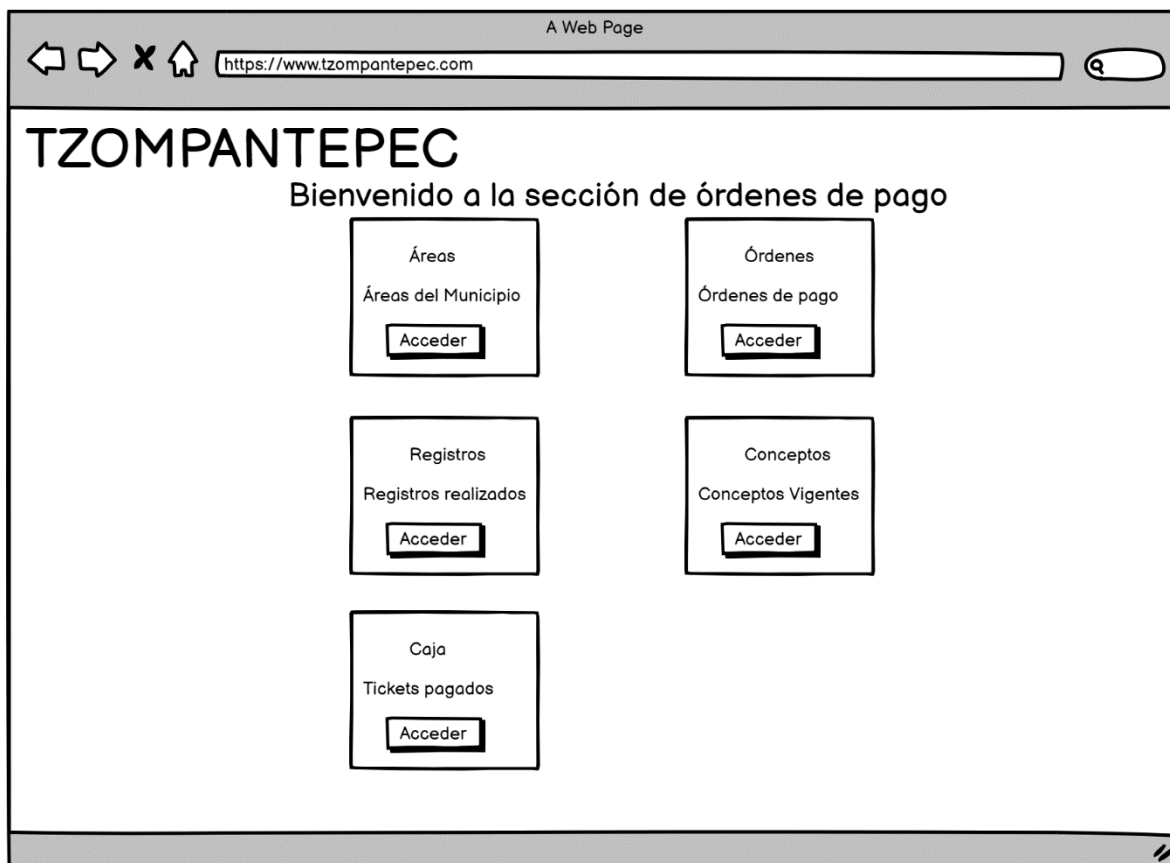


Figura 8. Ventana de opciones

En esta ventana se muestra el mismo encabezado, adicionalmente muestra que usuario ha iniciado sesión, posteriormente hay 5 fichas en las que se muestra las opciones que tiene disponible el usuario para ser utilizadas.

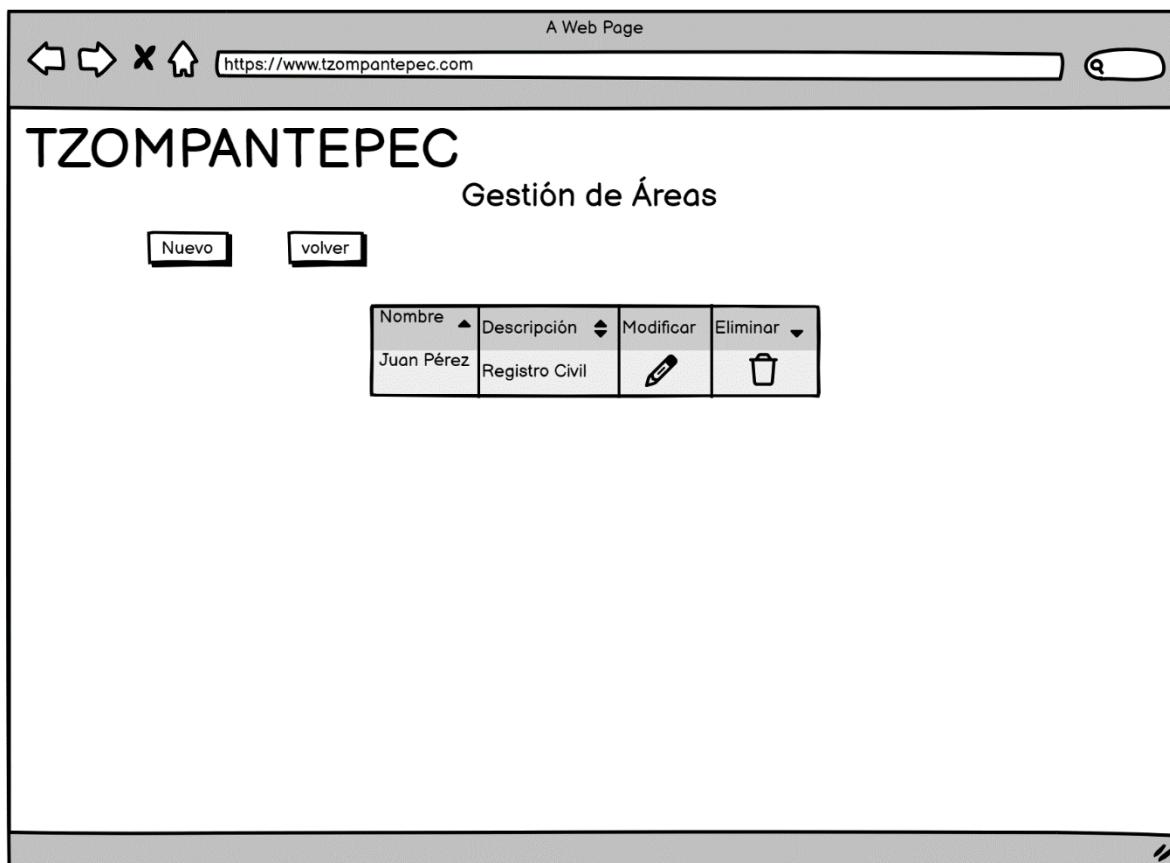


Figura 9. Registro de áreas

En esta ventana se muestra el encabezado con el nombre del municipio, y el nombre del usuario que ha iniciado sesión.

Tiene un título que pertenece a la gestión de áreas, posteriormente se muestran dos botones el de crear una nueva área y el de volver a la ventana de las tarjetas, y abajo se encuentra una tabla donde se muestran las áreas que ya han sido registradas con la opción de editar o eliminar cada una de ellas.

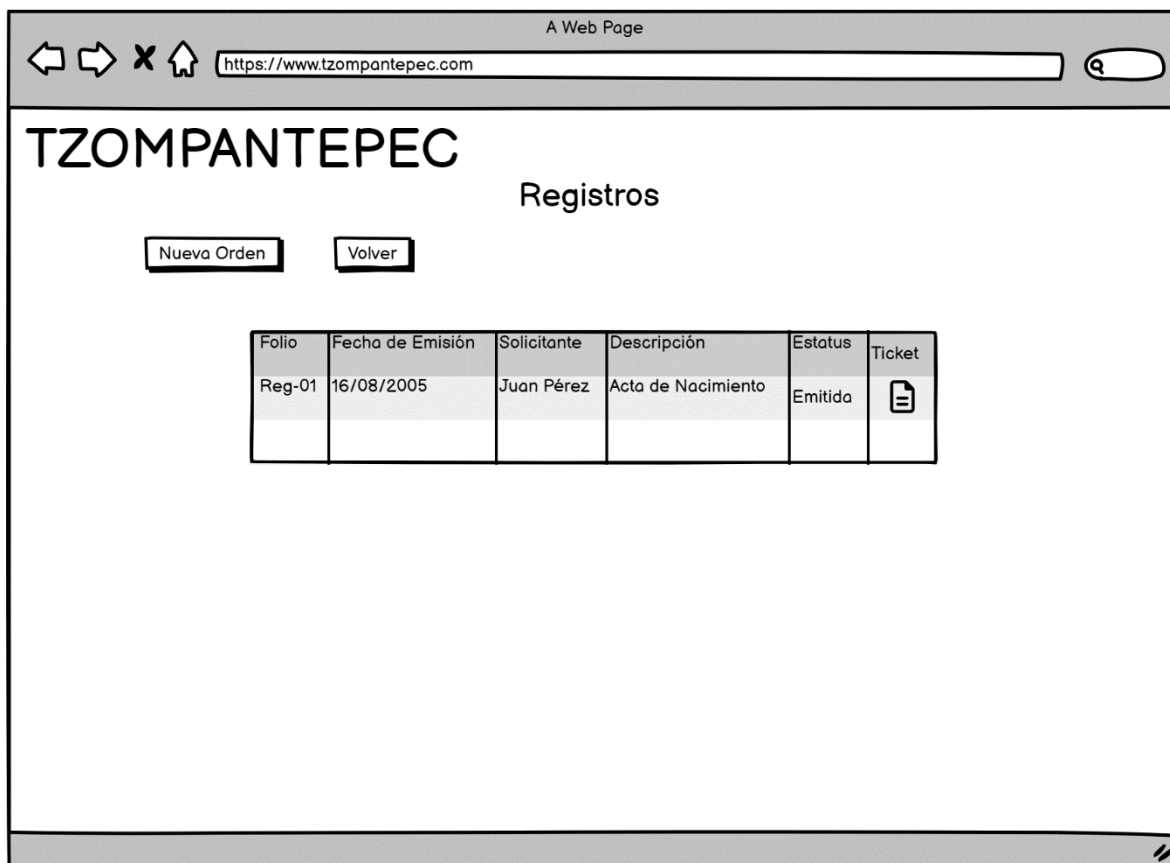


Figura 10. Ventana de Registros

En esta ventana se muestra el encabezado con el nombre del municipio, además del usuario que ha iniciado sesión,

Se muestra el título que pertenece a los registros, posteriormente dos botones con los cuales se puede crear una nueva orden o volver a la ventana anterior.

Se muestra una tabla que muestra las órdenes de pago que ya han sido generadas, con la opción de descargar un pdf del ticket de cada una de ellas.

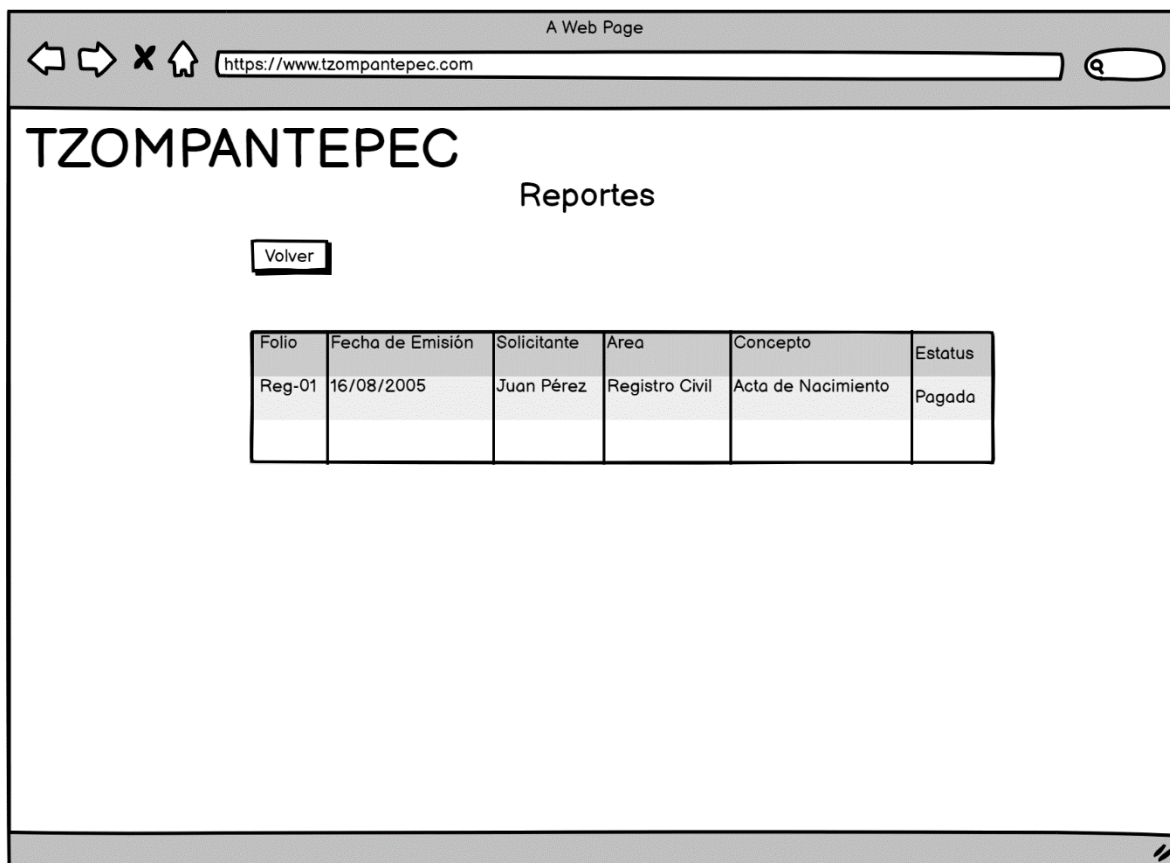


Figura 11. Ventana de reportes ordenes

Se muestra el encabezado con el nombre del municipio y del lado derecho el usuario que ha iniciado sesión.

Se muestra un botón que sirve para volver a la ventana anterior.

Se muestra una tabla la cual muestra las órdenes de pago que se han realizado, así mismo muestra un estatus de las que solo han sido emitidas y las que han sido pagadas, esta tabla tiene la opción de exportar a Excel, pdf o imprimir.

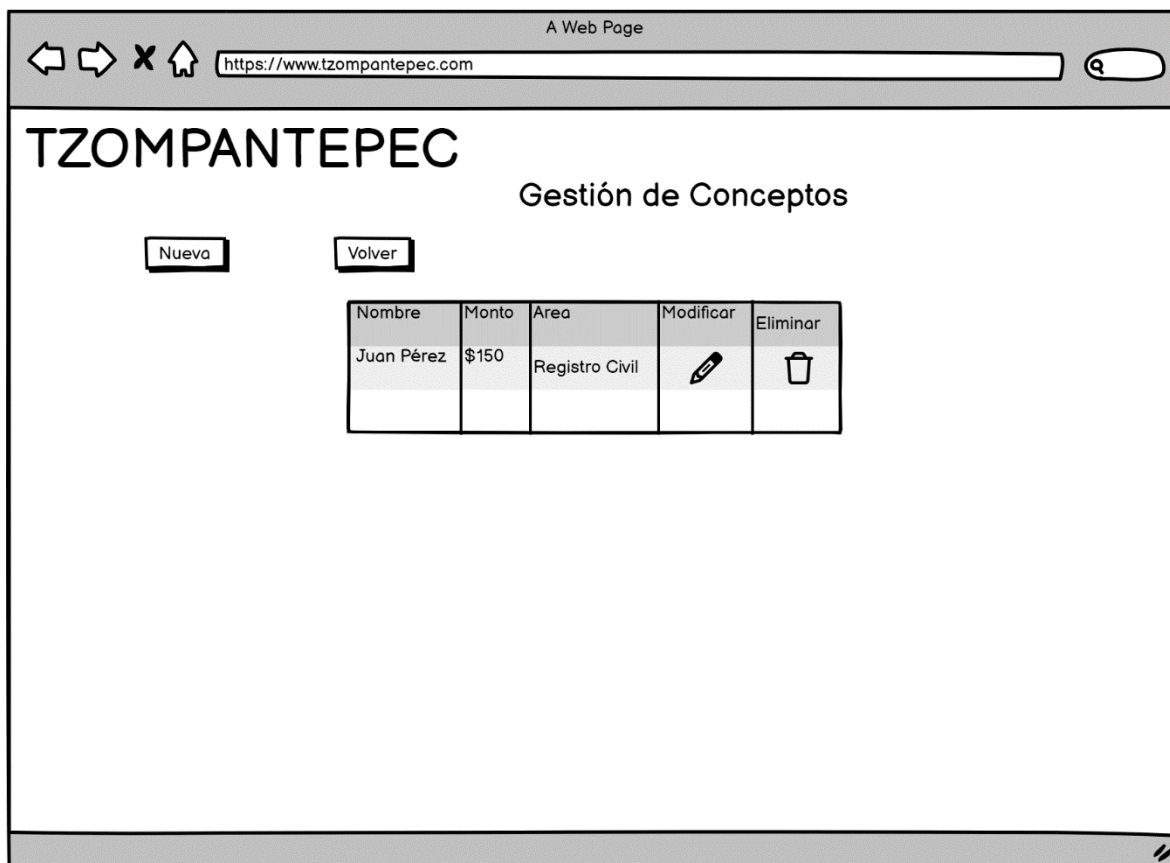


Figura 12. Ventana de gestión de conceptos

Se muestra el encabezado con el nombre del municipio y el nombre del usuario que ha iniciado sesión.

Se muestra el título que corresponde a la gestión de conceptos.

Se muestran dos botones uno para crear un nuevo concepto y el otro para volver a la pantalla anterior.

Se muestra una tabla con los conceptos que se han registrado con la opción de modificar o eliminar cada uno de ellos.

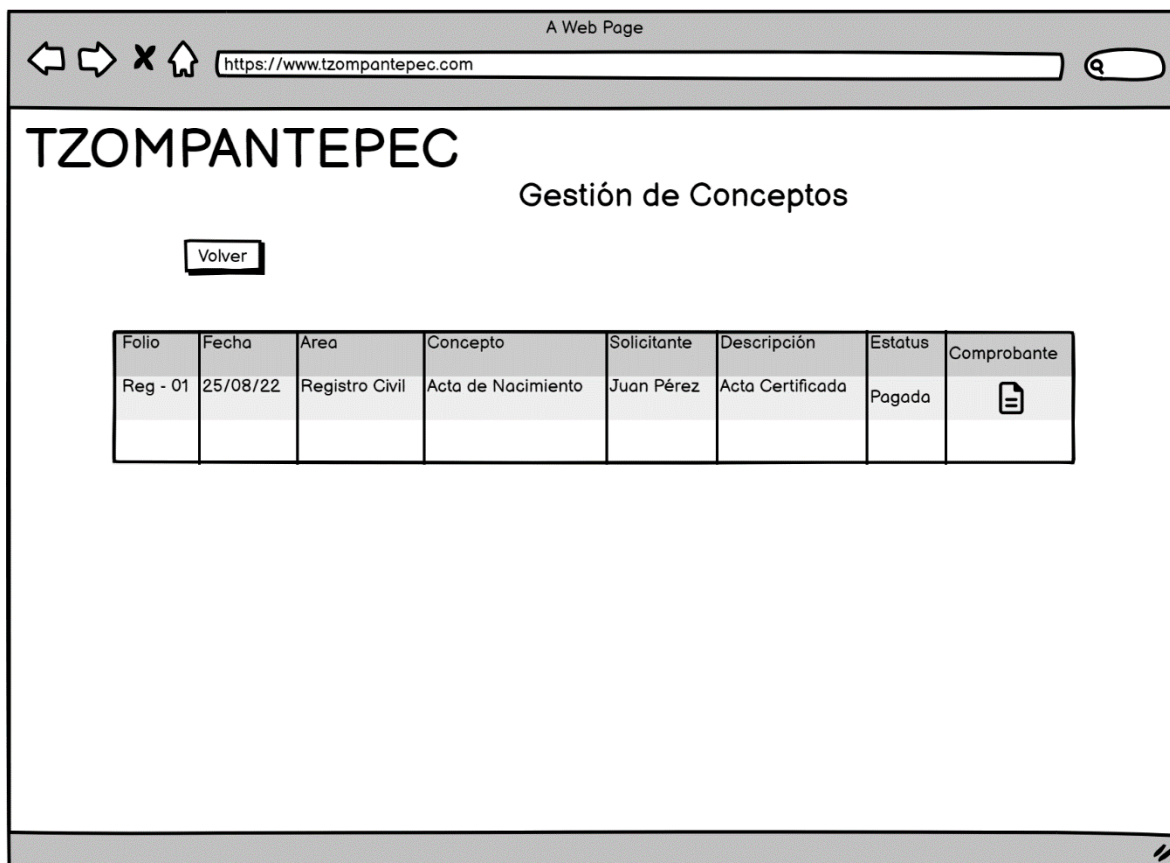


Figura 13. Ventana de listado de conceptos

Se muestra el encabezado, donde se muestra el nombre del municipio, además del lado superior derecho se muestra el usuario que ha iniciado sesión.

Se muestra un subtítulo donde se indica el nombre de la venta a la que pertenece es decir la gestión de conceptos.

Posterior se muestra un botón que el título de volver que sirve para regresar a la ventana anterior.

Y la tabla con los principales conceptos, además de las últimas dos columnas corresponden a indicar si esa orden ya ha sido pagada o no, y si es que si ha sido pagada en la siguiente columna aparece un icono para descargar el comprobante.

Proyecto en Django

Primero se debe de tener python instalado en la maquina, para el desarrollo de este proyecto se hizo uso de Visual Studio Code, se abre la terminal y creamos un entorno virtual con el cual se va a trabajar a lo largo del proyecto, tambien se realiza la instalacion de Django

Crear un entorno virtual – `python -m venv envtz`

Instalar Django –

`django-admin startproject ProyectoMunicipio`

Crear aplicación de Django –

`django-admin startapp appOpago`

Crear super usuario de Django –

`django-admin createsuperuser`

Crear migraciones de Django –

`python manage.py makemigrations`

Aplicar migraciones de Django –

`python manage.py migrate`

Iniciar el servidor de Django –

`python manage.py runserver`

```
settings.py M X
ProyectoMunicipio > settings.py > ...
23 # SECURITY WARNING: keep the secret key used in production secret!
24 SECRET_KEY = 'django-insecure-1xgd1=ywk-tubwq-9i91w+9psgtp#vxh+nn-8m135t9=r6c0jj'
25
26 # SECURITY WARNING: don't run with debug turned on in production!
27 DEBUG = True
28
29 ALLOWED_HOSTS = []
30
31
32 # Application definition
33
34 INSTALLED_APPS = [
35     'django.contrib.admin',
36     'django.contrib.auth',
37     'django.contrib.contenttypes',
38     'django.contrib.sessions',
39     'django.contrib.messages',
40     'django.contrib.staticfiles',
41     'appPredial',
42     'appUsuario',
43     'generales',
44     'appAgua',
45     'appOpago',
46     'crispy_forms',
47     'multiselectfield'
48
49 ]
50
51
```

Figura 14. Aplicación instalada en Django

Se crea la aplicación que corresponde a las órdenes de pago, así mismo se agrega a las aplicaciones instaladas en las configuraciones de Django

```
89 DATABASES = {
90     'default': {
91         'ENGINE': 'django.db.backends.postgresql_psycopg2',
92         'NAME': 'PruebaPredial',
93         'USER': 'postgres',
94         'PASSWORD': 'd_HgxyC9',
95         'HOST': 'localhost',
96         'PORT': '5432'
97     },
98     'ENGINE': 'django.db.backends.sqlite3',
99     'NAME': BASE_DIR / 'db.sqlite3',
100 }
```

Figura 15. Configuración de base de datos

Se configura la base de datos con la que se va a trabajar, por defecto Django al crear el proyecto trabaja con SQLite, pero se pueden configurar otras bases de datos como lo es PostgreSQL

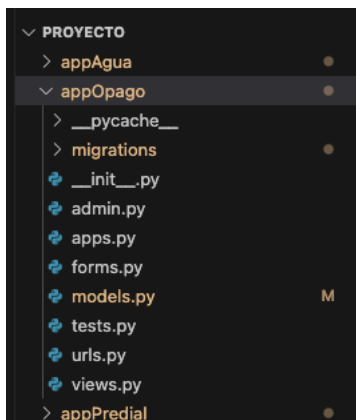


Figura 16. Estructura de la aplicación

Al crear el proyecto y la aplicación esta es la estructura que va a tener

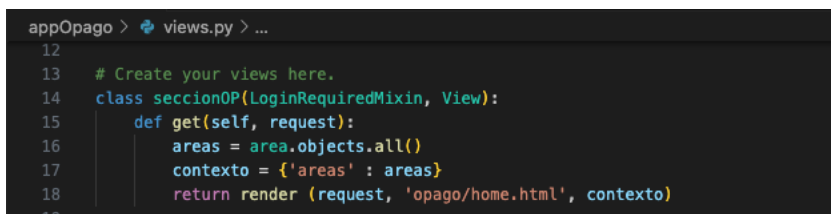


Figura 17. Vista de sección órdenes de pago

Se crea la primera vista que corresponde al inicio donde se muestran las funcionalidades que tiene la aplicación

```

19
20 def listarArea(request,):
21     areas = area.objects.all()
22     contexto = {'areas' : areas}
23     return render (request, 'opago/listar_area.html', contexto)
24
25 def crearArea (request):
26     if request.method == 'POST':
27         form = AreaForm(request.POST)
28         if form.is_valid():
29             form.save()
30             #messages.success(request, 'Registro Guardado')
31             return redirect('listar_area')
32     else:
33         form = AreaForm()
34     return render(request, 'opago/crear_area.html', {'form' : form})
35
36 def editarArea(request,id):
37     areas = area.objects.get(id=id)
38     if request.method == 'GET':
39         form = AreaForm(instance=areas)
40     else:
41         form=AreaForm(request.POST, instance=areas)
42         if form.is_valid():
43             form.save()
44             return redirect ('listar_area')
45     return render (request, 'opago/editar_area.html', {'form' : form})
46
47 def eliminarArea(request,id):
48     areas = area.objects.get(id=id)
49     if request.method == 'POST':
50         areas.delete()
51         return redirect ('listar_area')
52     return render (request, 'opago/eliminar_area.html', {'areas' : areas})
53

```

Figura 18. Vistas crud para áreas

Se crea las vistas que corresponden al área, desde listarla, crearla, editarla, y eliminarla cada vista renderiza una plantilla diferente para cumplir con la funcionalidad de un crud

```

54 class listOrden(ListView):
55     model = ordenes_pago
56     template_name = 'opago/consulta_area.html'
57
58 class createOrden(LoginRequiredMixin, CreateView, SuccessMessageMixin):
59     model = ordenes_pago
60     form_class = OPForm
61     template_name = 'opago/nueva_orden.html'
62     success_url = reverse_lazy('consulta_area')
63

```

Figura 19. Vista de listar y crear orden de pago

Estas vistas pertenecen a listar una orden de pago y crear una orden de pago, aquí hacemos uso de los mixin que tiene Django integrados para indicarle que el usuario debe haber iniciado sesión para poder acceder a la vista

```

64 class ticket(LoginRequiredMixin, View):
65     def get(self, request, *args, **kwargs):
66         template = get_template('opago/ticket.html')
67         context = {
68             'ordenes_pago': ordenes_pago.objects.get(pk=self.kwargs['pk']),
69             'comp': {'name': 'H. Ayuntamiento de Tzompantepec', 'ruc': '2414152315', 'address': 'Centro, 90490 Tzompantepec, Tlax.', 'titulo': 'Ticket de pago'},
70         }
71         html = template.render(context)
72         response = HttpResponse(content_type='application/pdf')
73         response['content-Disposition'] = 'attachment; filename="Ticket.pdf"'
74         pisa_status = pisa.CreatePDF(
75             html, dest=response
76         )
77         # if error then show some funny view
78         if pisa_status.err:
79             return HttpResponse('We had some error <pre>' + html + '</pre>')
80         return response
81
82     def list_history(request):
83         orden = ordenes_pago.objects.all()
84         contexto = {'orden': orden}
85         return render(request, 'opago/reportes.html', contexto)
86

```

Figura 20. Vista de pdf de ticket

Esta vista hace referencia a crear el ticket cuando se realiza una orden de pago, se utiliza el mixin de Django para que el usuario este logueado pueda descargar el ticket, se indica el contexto con el prellenado que va a tener el pdf generado

```

87 def listarConcepto(request):
88     conceptos = concepto.objects.all()
89     contexto = {'conceptos': conceptos}
90     return render(request, 'opago/listar_concepto.html', contexto)
91
92 def crearConcepto(request):
93     if request.method == 'POST':
94         form = conceptoForm(request.POST)
95         if form.is_valid():
96             form.save()
97             #messages.success(request, '¡Concepto Registrado')
98             return redirect('listar_conceptos')
99     else:
100         form = conceptoForm()
101         return render(request, 'opago/crear_concepto.html', {'form': form})
102
103 def editarConcepto(request, id):
104     conceptos = concepto.objects.get(id=id)
105     if request.method == 'GET':
106         form = conceptoForm(instance=conceptos)
107     else:
108         form = conceptoForm(request.POST, instance=conceptos)
109         if form.is_valid():
110             form.save()
111             return redirect('listar_conceptos')
112     return render(request, 'opago/editar_concepto.html', {'form': form})
113
114 def eliminarConcepto(request, id):
115     conceptos = concepto.objects.get(id=id)
116     if request.method == 'POST':
117         conceptos.delete()
118         return redirect('listar_conceptos')
119     return render(request, 'opago/eliminar_concepto.html', {'conceptos': conceptos})
120

```

Figura 21. Vistas crud para concepto

Estas vistas hacen referencia a los conceptos, las cuales cumplen la función de un crud para interactuar a través de las plantillas

```

121 def reportes(request):
122     orden = ordenes_pago.objects.all()
123     contexto = {'orden': orden}
124     return render(request, 'opago/reportes.html', contexto)
125

```

Figura 22. Vista para lista de reportes

Esta vista hace referencia a listar los reportes que se han generado a través de las órdenes de pago

```

126 class estatus (LoginRequiredMixin, ListView):
127     model = ordenes_pago
128     template_name = 'opago/caja.html'
129
130 class updateEstatus (LoginRequiredMixin, UpdateView):
131     model = ordenes_pago
132     form_class = respuestaForm
133     template_name = 'opago/modificar.html'
134     success_url = reverse_lazy('caja')
135

```

Figura 23. Vista de listar y actualizar estatus

Esta vista hace referencia a visualizar el estatus que tiene una orden de pago, y también se puede realizar la actualización del estatus

```

136 class pdfView(LoginRequiredMixin, View):
137     def get(self, request, *args, **kwargs):
138         template = get_template('opago/comprobante.html')
139         context = {
140             'ordenes_pago': ordenes_pago.objects.get(pk=self.kwargs['pk']),
141             'comp': {'name': 'H. Ayuntamiento de Tzompantepec', 'ruc': '2414152315',
142                     'address': 'Centro, 90490 Tzompantepec, Tlax.'},
143         }
144         html = template.render(context)
145         response = HttpResponse(content_type='application/pdf')
146         response['content-Disposition'] = 'attachmente; filename="comprobante.pdf"'
147         pisa_status = pisa.CreatePDF(
148             html, dest=response
149         )
150         # if error then show some funny view
151         if pisa_status.err:
152             return HttpResponse('we had some errors <pre>' + html + '</pre>')
153         return response

```

Figura 24. Vista crear pdf de comprobante

Cuando la orden ha sido pagada se puede descargar un pdf el cual figura como comprobante, este se llena con datos prellenados más los datos de la orden de pago

```
models.py M X
appOpago > models.py > concepto
1 from django.db import models
2 from datetime import datetime
3 from time import strftime
4
5 # Create your models here.
6 class area(models.Model):
7     nombre = models.CharField(max_length=50, null=True, blank=True)
8     descripcion = models.CharField(max_length=50, null=True, blank=True)
9
10     def __str__(self):
11         return '%s' % (self.nombre)
12
13     class Meta:
14         verbose_name = 'Area'
15         verbose_name_plural = 'Area'
16
17 class concepto(models.Model):
18     nombre = models.CharField(max_length=30)
19     monto = models.CharField(default=0, max_length=10)
20     area = models.ForeignKey(area, on_delete=models.CASCADE, default='1')
21
22     def __str__(self):
23         return '%s' % (self.nombre)
24
25 class productos(models.Model):
26     producto = models.CharField(max_length=30)
27
28     def __str__(self):
29         return '%s' % (self.producto)
30
31 def clave():
32     mes = datetime.now().month
33     mes = strftime('%b')
34     op = ordenes_pago.objects.count() + 1
35     return f'{mes}-{op}'
36
37 class ordenes_pago(models.Model):
38     id = models.AutoField(primary_key=True)
39     folio = models.CharField(max_length=50, default=clave)
40     fecha_emision = models.DateField(default=datetime.now)
41     nombre_solicitante = models.CharField(max_length=100)
42     descripcion = models.CharField(max_length=100)
43     estatus = models.BooleanField(default=False)
44     producto = models.FileField(upload_to='media/', null=True, blank=True)
45     # concepto = models.CharField(max_length=100)
46     monto = models.FloatField(default=0)
47     area = models.ForeignKey(area, on_delete=models.CASCADE, default='1')
48     concepto = models.ForeignKey(concepto, on_delete=models.CASCADE, default='1')
49
50     def __str__(self):
51         return '%s' % self.id
52
53     def save(self, *args, **kwargs):
54
55         return '%s' % self.id
56
57     def save(self, *args, **kwargs):
58         if self.concepto:
59             self.monto = self.concepto.monto
60             super().save(*args, **kwargs)
61
62     class meta:
63         db_table = 'ordenes_pago'
```

Figura 25. Modelos para base de datos

Estos son los modelos que pertenecen a toda la aplicación, cada modelo tiene sus atributos correspondientes haciendo uso de llaves primarias y otros atributos

```
admin.py X
appOpago > admin.py
1 from django.contrib import admin
2 from .models import area, ordenes_pago, concepto
3
4 # Register your models here.
5
6 admin.site.register(area)
7 admin.site.register(ordenes_pago)
8 admin.site.register(concepto)
```

Figura 26. Modelos en el administrador de Django

Se registran los modelos que se han realizado en el admin.py para poder visualizarlos en el administrador de Django

```

forms.py
1 from django import forms
2 from django.forms.models import fields_for_model
3 from .models import area, ordenes_pago, concepto
4
5 class AreaForm(forms.ModelForm):
6     class Meta:
7         model = area
8         fields = '__all__'
9
10    def __init__(self, *args, **kwargs):
11        super().__init__(*args, **kwargs)
12        for field in iter(self.fields):
13            self.fields[field].widget.attrs.update({'class': 'form-control'})
14
15 class OPForm(forms.ModelForm):
16     class Meta:
17         model = ordenes_pago
18         fields = [
19             'folio',
20             'fecha_emision',
21             'nombre_solicitante',
22             'descripcion',
23             'area',
24             'concepto',
25             'monto',
26             'producto',
27         ]
28
29     labels = {
30         'folio': 'Folio',
31         'fecha_emision': 'Fecha de emision',
32         'nombre_solicitante': 'Nombre del solicitante',
33         'descripcion': 'Descripcion',
34         'area': 'Area',
35         'concepto': 'Concepto',
36         'monto': 'Monto',
37         'producto': 'Producto'
38     }
39
40     widget = {
41         'folio': forms.TextInput(),
42         'fecha_emision': forms.DateInput(attrs={'type': 'date', 'class': 'form-control'}),
43         'nombre_solicitante': forms.TextInput(),
44         'descripcion': forms.TextInput(),
45         'area': forms.TextInput(),
46         'concepto': forms.TextInput(),
47         'monto': forms.TextInput(),
48         'producto': forms.FileInput()
49     }
50
51    def __init__(self, *args, **kwargs):
52        super().__init__(*args, **kwargs)
53        for field in iter(self.fields):
54
forms.py
51    def __init__(self, *args, **kwargs):
52        super().__init__(*args, **kwargs)
53        for field in iter(self.fields):
54            self.fields[field].widget.attrs.update({'class': 'form-control'})
55
56 class ConceptForm(forms.ModelForm):
57     class Meta:
58         model = concepto
59         fields = ['nombre', 'monto', 'area']
60
61    def __init__(self, *args, **kwargs):
62        super().__init__(*args, **kwargs)
63        for field in iter(self.fields):
64            self.fields[field].widget.attrs.update({'class': 'form-control'})
65
66 class respuestaForm(forms.ModelForm):
67     status = [
68         (True, 'Pagado'),
69         (False, 'Por pagar')
70     ]
71     estatus = forms.ChoiceField(choices=status, widget=forms.RadioSelect)
72
73     class Meta:
74         model = ordenes_pago
75         fields = [
76             'id',
77             'estatus',
78         ]
79
80     labels = {
81         'id': 'id',
82         'estatus': 'estatus',
83     }

```

Figura 27. Formularios de Django

Se crean los formularios para renderizar las plantillas, aquí indicamos los campos que van a tener alguna validación por ejemplo las fechas, además de cómo se va a renderizar el campo en las plantillas y no se pinte como llamamos el campo en el modelo

```

urls.py
65 path("", include(('appUsuario.urls', 'usuarios'))),
66 path('login/', Log.as_view(), name='login'),
67
68 #URL de Aplicacion ordenes de pago
69 path('', include('appOpago.urls')),
70 ]
71

```

Figura 28. URL en la aplicación principal

Para poder acceder desde el navegador a la aplicación que se ha creado, hay que registrar la url de la aplicación en el archivo urls.py de la aplicación en general

```
urls.py x
appOpago > urls.py > ...
4 from appOpago.views import seccionOP, listOrden, createOrden, ticket, estatus, updateEstatus, pdfView
5
6 urlpatterns = [
7     path('ordenes/', seccionOP.as_view(), name='ordenesp'),
8     path('listar_areas/', views.listarArea, name='listar_area'),
9     path('crear_area/', views.crearArea, name='crear_area'),
10    re_path(r'^editar_area/(?P<id>\d+)$', views.editarArea, name="editar_area"),
11    re_path(r'^eliminar_area/(?P<id>\d+)$', views.eliminarArea, name="eliminar_area"),
12    path('consulta_area/', listOrden.as_view(), name='consulta_area'),
13    path('nueva_orden/', createOrden.as_view(), name='nueva_orden'),
14    path('ticket/<int:pk>', ticket.as_view(), name='ticket'),
15    path('listar_conceptos/', views.listarConcepto, name='listar_conceptos'),
16    path('crear_concepto/', views.crearConcepto, name='crear_concepto'),
17    re_path(r'^editar_concepto/(?P<id>\d+)$', views.editarConcepto, name="editar_concepto"),
18    re_path(r'^eliminar_concepto/(?P<id>\d+)$', views.eliminarConcepto, name="eliminar_concepto"),
19    path('reportes/', views.reportes, name='reportes'),
20    path('caja/', estatus.as_view(), name='caja'),
21    path('modificar/<int:pk>', updateEstatus.as_view(), name='modificar'),
22    path('pdf/<int:pk>', pdfView.as_view(), name='pdf'),
23 ]
```

Figura 29. URLs de la aplicación órdenes de pago

En el archivo urls.py de la aplicación se registran todas las que va a tener la aplicación

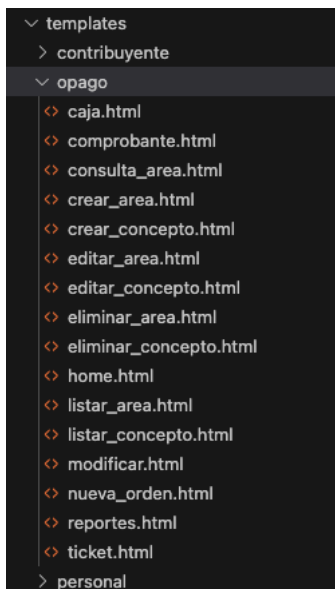


Figura 30. Plantillas de la aplicación

En la carpeta de “templates” y una subcarpeta de “opago” se crean todas las plantillas de cada una de las vistas que se crearon

```
{% block head %}
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no" />
  <meta name="description" content="" />
  <meta name="author" content="" />
  <title>Freelancer - Start Bootstrap Theme</title>
  <!-- Favicon -->
  <link rel="icon" type="image/x-icon" href="assets/favicon.ico" />
  <!-- Font Awesome icons (free version) -->
  <script src="https://use.fontawesome.com/releases/v5.15.4/js/all.js" crossorigin="anonymous"></script>
  <!-- Google fonts -->
  <link href="https://fonts.googleapis.com/css?family=Montserrat:400,700" rel="stylesheet" type="text/css" />
  <link href="https://fonts.googleapis.com/css?family=Lato:400,700,400italic,700italic" rel="stylesheet" type="text/css" />
  <!-- Estilo del botón para agregar, buscar contribuyente, predio, etc -->
  <link href="{% static 'css/botones.css' %}" rel="stylesheet" />
  <!-- Core theme CSS (includes Bootstrap) -->
  <link href="{% static 'css/styles.css' %}" rel="stylesheet" />
  <link href="{% static 'css/styles2.css' %}" rel="stylesheet" />
  <link href="{% static 'css/styles-personal.css' %}" rel="stylesheet">
  <link href="{% static 'css/messages.css' %}" rel="stylesheet">
  <link href="{% static 'css/menu.css' %}" rel="stylesheet">

  <!-- Font Awesome -->
  <link rel="stylesheet" href="{% static 'plugins/fontawesome-free/css/all.min.css' %}">
  <!-- DataTables -->
  <link rel="stylesheet" href="{% static 'plugins/datatables-bs4/css/dataTables.bootstrap4.min.css' %}">
  <link rel="stylesheet" href="{% static 'plugins/datatables-responsive/css/responsive.bootstrap4.min.css' %}">
  <link rel="stylesheet" href="{% static 'plugins/datatables-buttons/css/buttons.bootstrap4.min.css' %}">

{% block datarangepicker%}

{% endblock%}
```

Figura 31. Inserción de CSS y tipo de letra

En la plantilla base principalmente en el head se agregan los css, los iconos, el tipo de letra y los css de las datatables

```
<!-- DataTables & Plugins -->
<script src="{% static 'plugins/datatables/jquery.dataTables.min.js' %}"></script>
<script src="{% static 'plugins/datatables-bs4/js/dataTables.bootstrap4.min.js' %}"></script>
<script src="{% static 'plugins/datatables-responsive/js/dataTables.responsive.min.js' %}"></script>
<script src="{% static 'plugins/datatables-responsive/js/responsive.bootstrap4.min.js' %}"></script>
<script src="{% static 'plugins/datatables-buttons/js/dataTables.buttons.min.js' %}"></script>
<script src="{% static 'plugins/datatables-buttons/js/buttons.bootstrap4.min.js' %}"></script>
<script src="{% static 'plugins/jszip/jszip.min.js' %}"></script>
<script src="{% static 'plugins/pdfmake/pdfmake.min.js' %}"></script>
<script src="{% static 'plugins/pdfmake/vfs_fonts.js' %}"></script>
<script src="{% static 'plugins/datatables-buttons/js/buttons.html5.min.js' %}"></script>
<script src="{% static 'plugins/datatables-buttons/js/buttons.print.min.js' %}"></script>
<script src="{% static 'plugins/datatables-buttons/js/buttons.colVis.min.js' %}"></script>
```

Figura 32. Datatables y plugins en el footer

En el footer se insertan los scripts de las datatables y plugis ya que de esta plantilla van a extender todas las demás

```

<!-- Ion-Icons theme JS-->
<script type="module" src="https://unpkg.com/ionicons@5.5.2/dist/ionicons/ionicons.esm.js"></script>
<script nomodule src="https://unpkg.com/ionicons@5.5.2/dist/ionicons/ionicons.js"></script>

```

Figura 33. Inserción de iconos

Se agregan los scripts de los iconos que se van a utilizar en las plantillas

```

<!-- Sweetalert-->
<script src="//cdn.jsdelivr.net/npm/sweetalert2@11"></script>

<script>
  {% if messages %}
    var html = '<p>';
    {% for message in messages %}
      html += '{{ message }}<br>';
    {% endfor %}
    html += '</p>'
    Swal.fire({
      title: 'Error!',
      html: html,
      icon: 'error',
      allowOutsideClick: false,
      clickOutsideToClose: false,
      confirmButtonText: 'Aceptar'
    })
  {% endif %}
</script>

<!-- Funciones JS Ajax-->
<script src="{% static 'js/functions.js' %}"></script>

```

Figura 34. Inserción de mensajes de alerta

El Script de sweetalert son los mensajes que se muestran cuando se realiza un proceso exitosamente o hay alguna alerta que mostrar al usuario

```

<!-- Page specific script -->
<script>
    $(function() {
        $("#example1").DataTable({
            "responsive": true,
            "lengthChange": false,
            "autoWidth": false,
            "buttons": ["copy", "csv", "excel", "pdf", "print", "colvis"]
        }).buttons().container().appendTo('#example1_wrapper .col-md-6:eq(0)');
        $('#example2').DataTable({
            "paging": true,
            "lengthChange": false,
            "searching": true,
            "ordering": true,
            "info": true,
            "autoWidth": false,
            "responsive": true,
        });
        $('#example3').DataTable({
            "paging": true,
            "lengthChange": false,
            "searching": true,
            "ordering": true,
            "info": true,
            "autoWidth": false,
            "responsive": true,
        });
        $('#example4').DataTable({
            "responsive": true,
            "lengthChange": false,
            "autoWidth": false,
            "buttons": ["copy", "csv", "excel", "pdf", "print", "colvis"]
        }).buttons().container().appendTo('#example1_wrapper .col-md-6:eq(0)');
    });
</script>

```

Figura 35. Script de datatables

Este script también pertenece a la datatables, pero aquí se indica que botones va a tener la tabla para poder exportar el contenido, además de otros atributos como el orden de los datos y si será responsiva

```

<> home.html x
templates > opago > <> home.html
1  {% extends 'personal/base.html' %}
2  {% load static %}
3
4  {% block contenido %}
5  > <div class="container container-data d-flex align-items-center flex-column">-
58 </div>
59  {% endblock contenido %}

```

Figura 36. Esqueleto de plantilla

Para crear cada una de las plantillas al inicio de la misma hacemos la extensión de la plantilla base, además de cargar los estáticos, y dentro del bloque de contenido todo lo que pertenece a su función

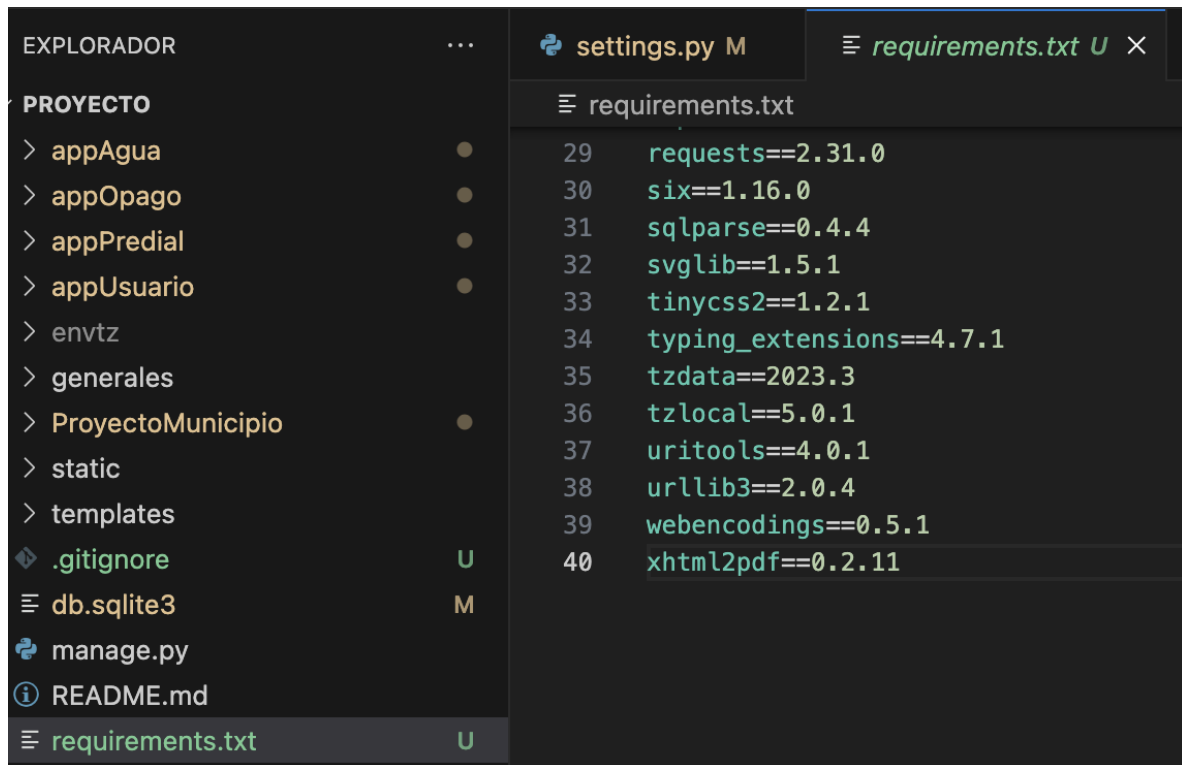


Figura 37. Requerimientos del entorno virtual

Se realiza un txt con los requerimientos para después realizar el despliegue del mismo o instalar en otro entorno virtual

Crear requerimientos –

Pip freeze > requirements.txt

Resultados

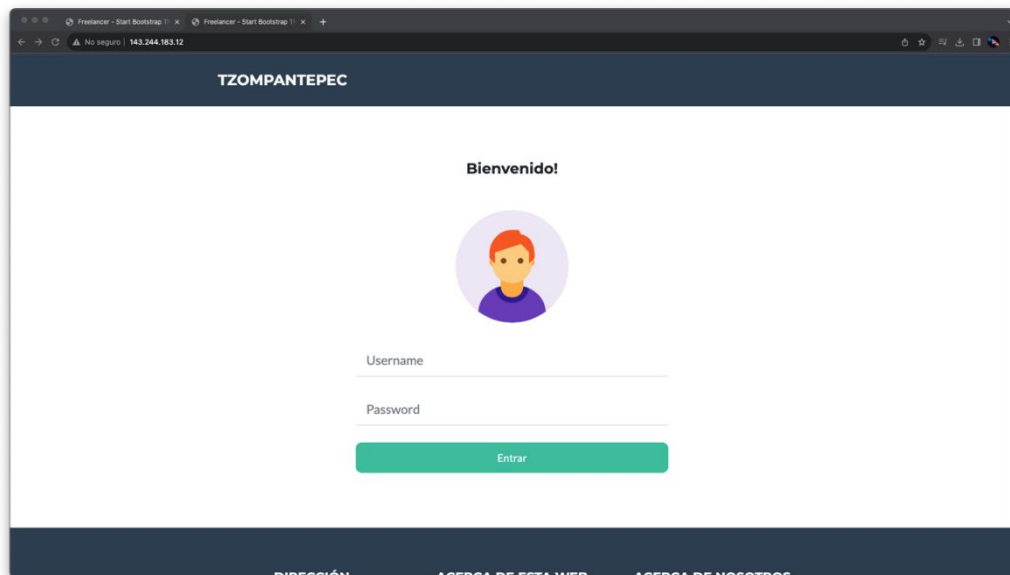


Figura 38. Inicio de sesión

Página de inicio, se muestra el inicio de sesión para introducir el correo y contraseña del usuario antes registrado



Figura 39. Módulos

Una vez iniciado sesión se muestran los módulos, el cual este proyecto pertenece a las órdenes de pago

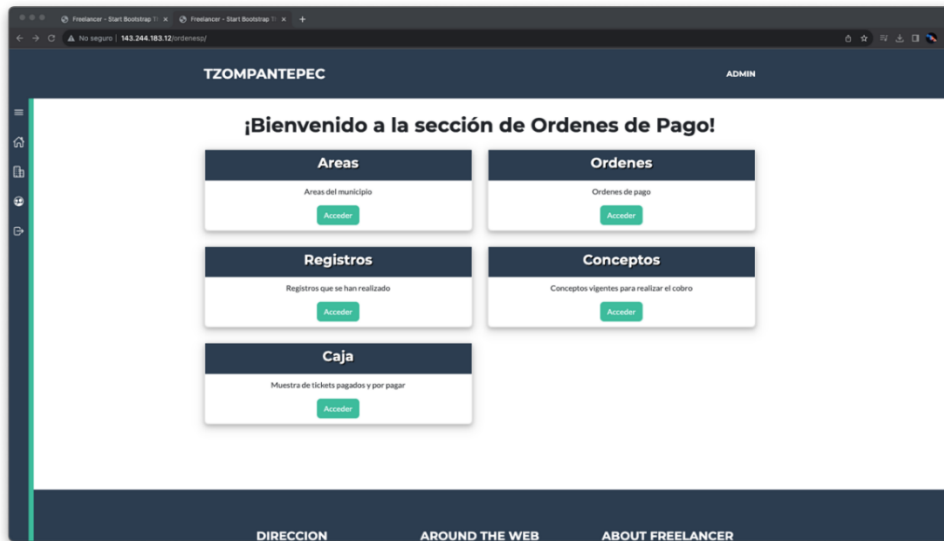


Figura 40. Sección de órdenes de pago

Este es el menú de órdenes de pago, del lado superior derecho se muestra el usuario que ha ingresado, este va a cambiar de acuerdo al usuario que haya ingresado, además se muestran los botones para acceder a cada una de las opciones disponibles

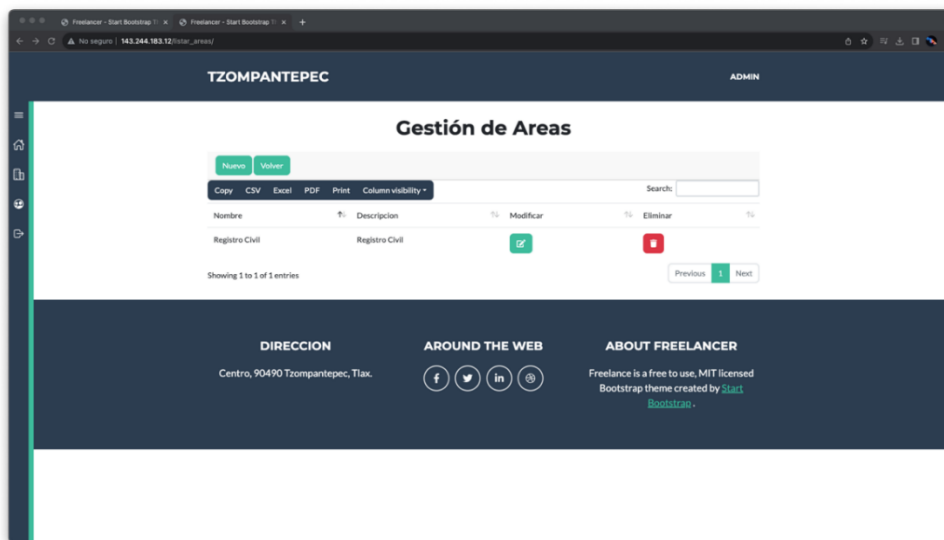


Figura 41. Gestión de áreas

La gestión de áreas corresponde a las que están disponibles en el municipio, esta tabla se puede exportar a diferentes formatos de archivo, además de imprimir, cada una de las áreas tiene la opción de ser modificada o eliminada

The screenshot shows a web browser window with the URL `143.244.183.12/crear_area/`. The page header includes 'TZOMPANTEPEC' and 'ADMIN'. A sidebar on the left contains navigation icons. The main content area features a magnifying glass icon over a document, followed by the title 'Nueva Area'. Below the title are two input fields: 'Nombre:' and 'Descripcion:'. At the bottom of the form are two green buttons: 'Crear' and 'Cancelar'. The footer contains three sections: 'DIRECCION' (Centro, 90490 Tzompantepec, Tlax.), 'AROUND THE WEB' (social media icons), and 'ABOUT FREELANCER' (Freelance is a free to use, MIT licensed Bootstrap theme created by [Start Bootstrap](#)).

Figura 42. Crear nueva área

Esta es la plantilla que se muestra al dar clic en el botón de nuevo para crear una nueva área, se rellenan los campos de nombre y descripción, para guardar está el botón de crear o para cancelar

The screenshot shows a web browser window with the URL `143.244.183.12/modificar_area/2`. The page header includes 'TZOMPANTEPEC' and 'ADMIN'. A sidebar on the left contains navigation icons. The main content area features a magnifying glass icon over a document, followed by the title 'Modificar Area'. Below the title are two input fields: 'Nombre:' (containing 'Registro Civil') and 'Descripcion:' (containing 'Registro Civil'). At the bottom of the form are two green buttons: 'Modificar' and 'Cancelar'. The footer contains three sections: 'DIRECCION' (Centro, 90490 Tzompantepec, Tlax.), 'AROUND THE WEB' (social media icons), and 'ABOUT FREELANCER' (Freelance is a free to use, MIT licensed Bootstrap theme created by [Start Bootstrap](#)).

Figura 43. Modificar área existente

Esta plantilla se muestra cuando el área se va a modificar, una vez terminado este botón de modificar para guardar el cambio o el botón de cancelar

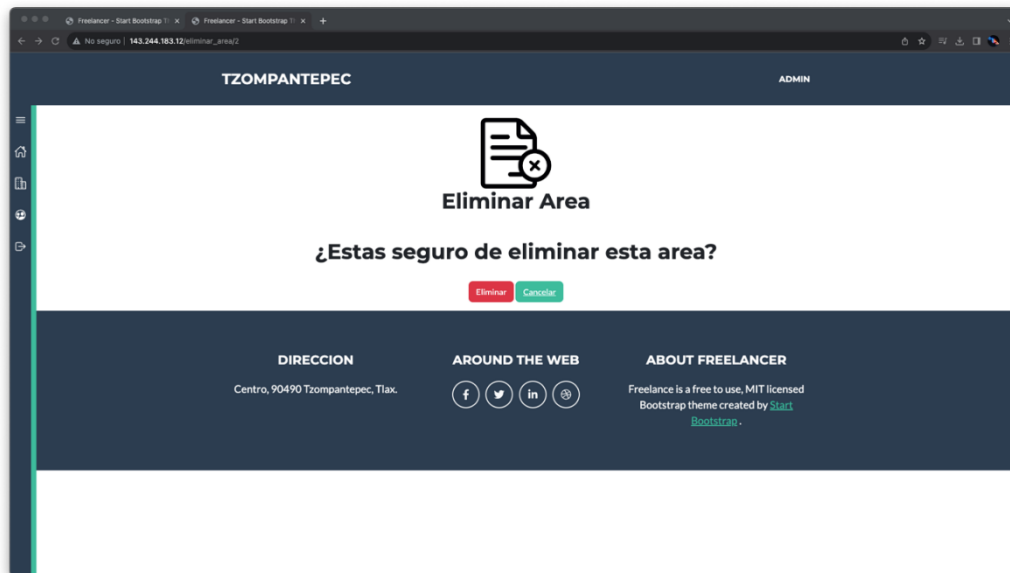


Figura 44. Eliminar área

Si se desea eliminar el área antes de realizar el proceso le pregunta al usuario si está seguro de realizar esta acción

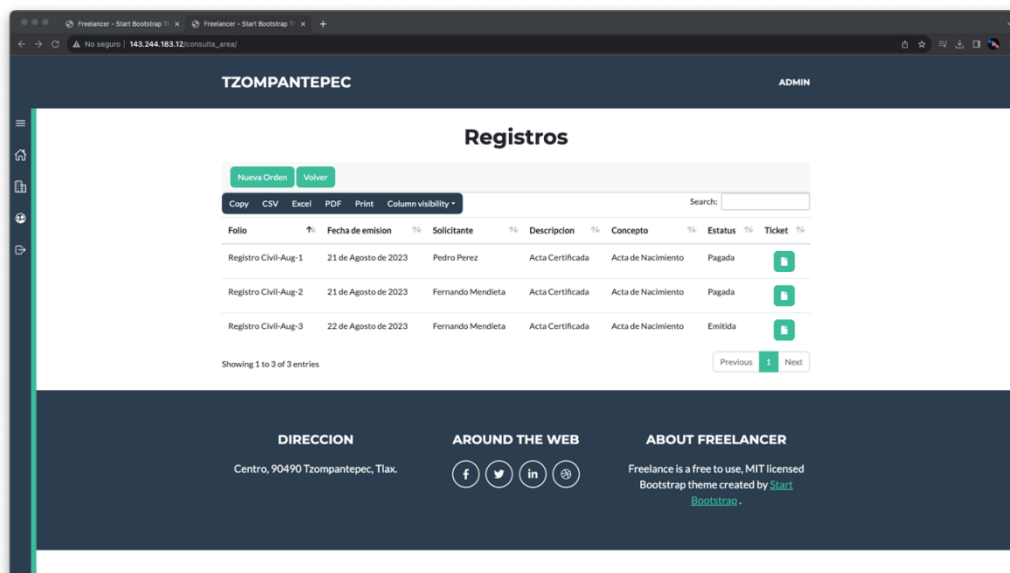


Figura 45. Listado de órdenes de pago

Se muestra el listado de las órdenes de pago realizadas, mostrando el estatus y el icono de generar el ticket

Figura 46. Crear nueva orden de pago

Se genera una nueva orden de pago, en automático al cargar la plantilla se obtiene el folio y la fecha de emisión, además de un listado de áreas y conceptos para elegir la orden de pago

Figura 47. Ubicación de guardado ticket

Si se desea descargar el ticket depende el navegador se muestra donde desea guardar el archivo o automáticamente se ira a su carpeta descargas

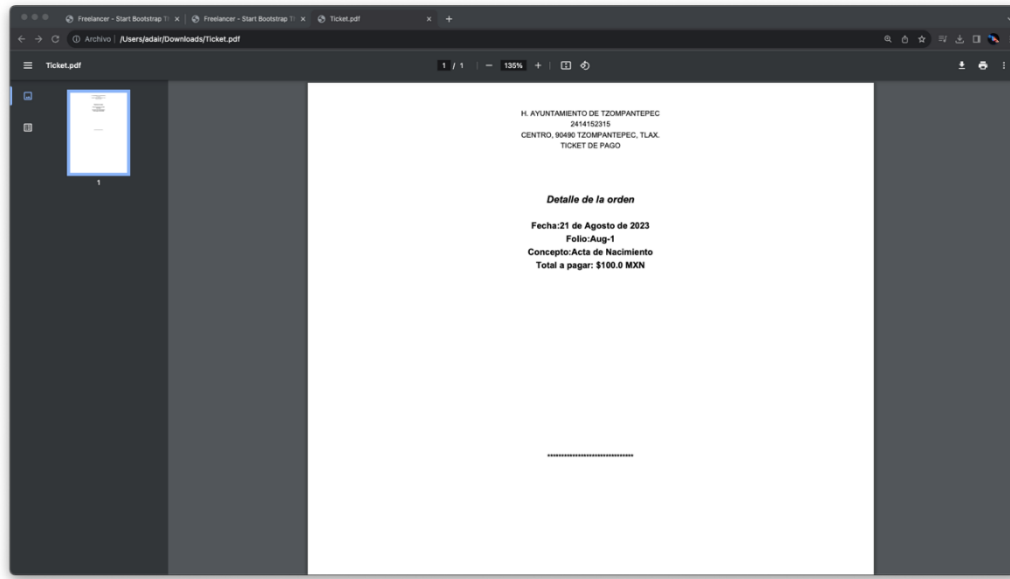


Figura 48. Ticket generado

Este es el ticket que se genera en formato pdf con los datos prellenados que se indicaron en la vista, además del detalle de la orden en el que fue creado

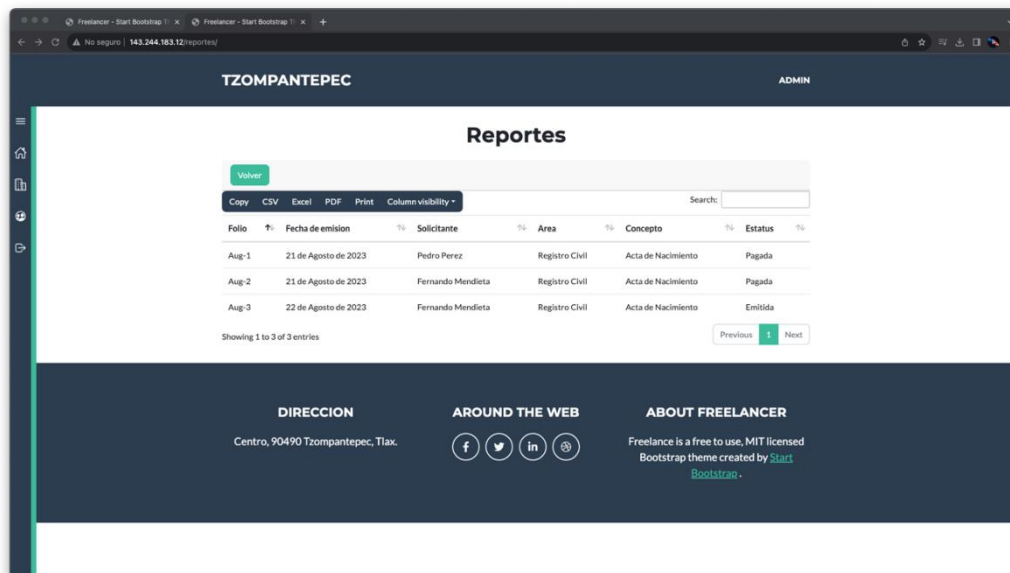


Figura 49. Reportes

Se muestra el listado de las órdenes de pago con su estatus para realizar un filtrado o exportar los resultados a un tipo de archivo



Figura 50. Listado de conceptos

Se muestra el listado de conceptos con la opción de exportar la tabla y modificar o eliminar cada concepto que se encuentra listado

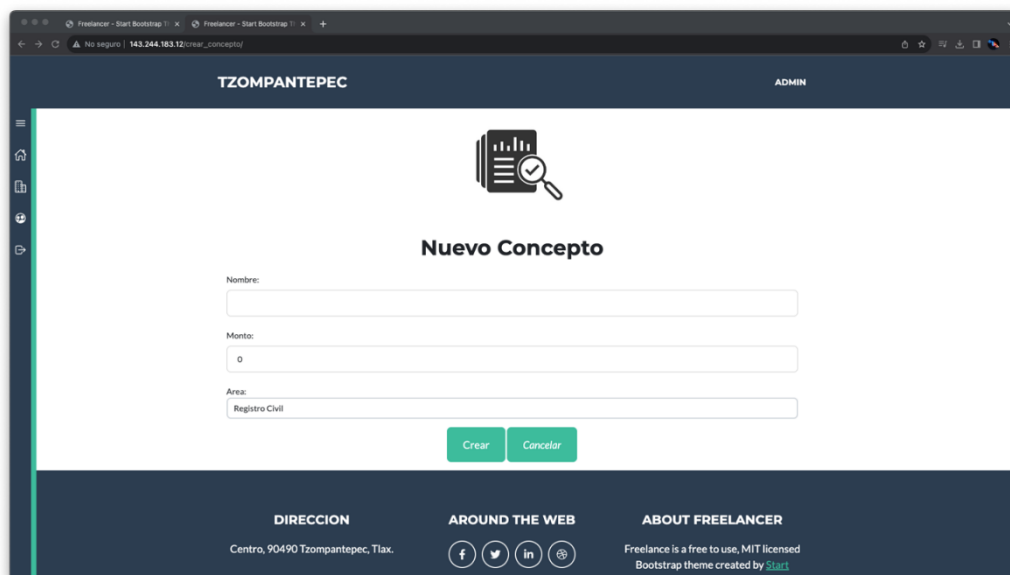


Figura 51. Crear nuevo concepto

Para crear un nuevo concepto se debe indicar el nombre del mismo, además del monto que le corresponde y a que área pertenece

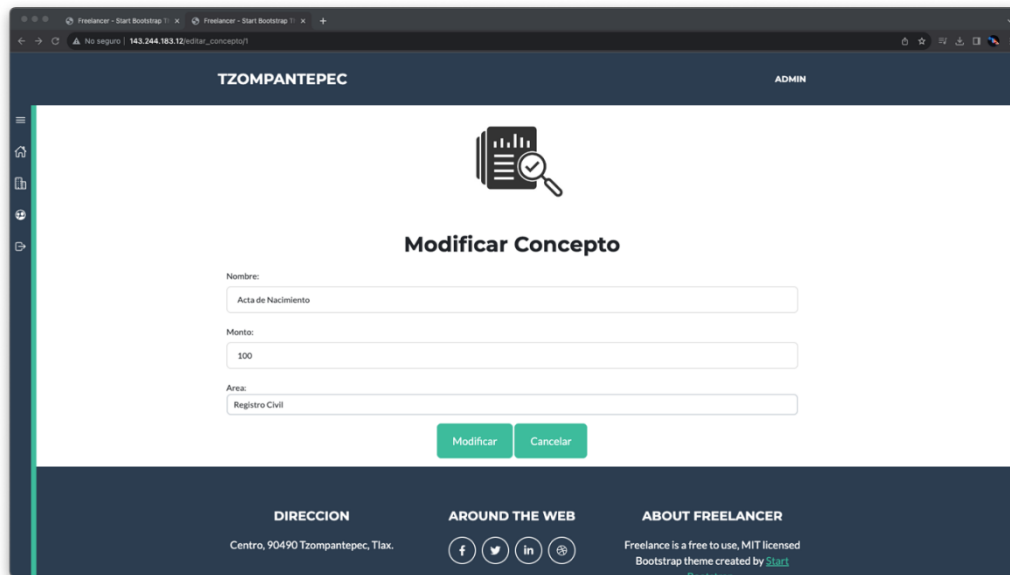


Figura 52. Modificar concepto actual

Si hay algún dato que hay que corregir al concepto esta la opción de modificar el cual va a cargar los campos que ha rellenado para ser modificados

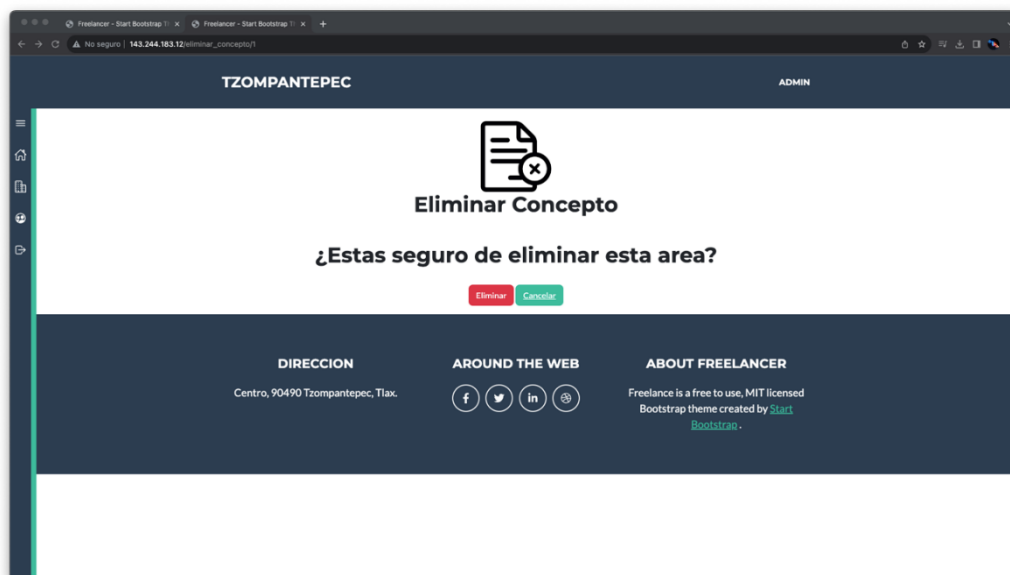


Figura 53. Eliminar un concepto existente

Para eliminar cada uno de los conceptos el sistema le indicara que si está seguro de realizar esta acción

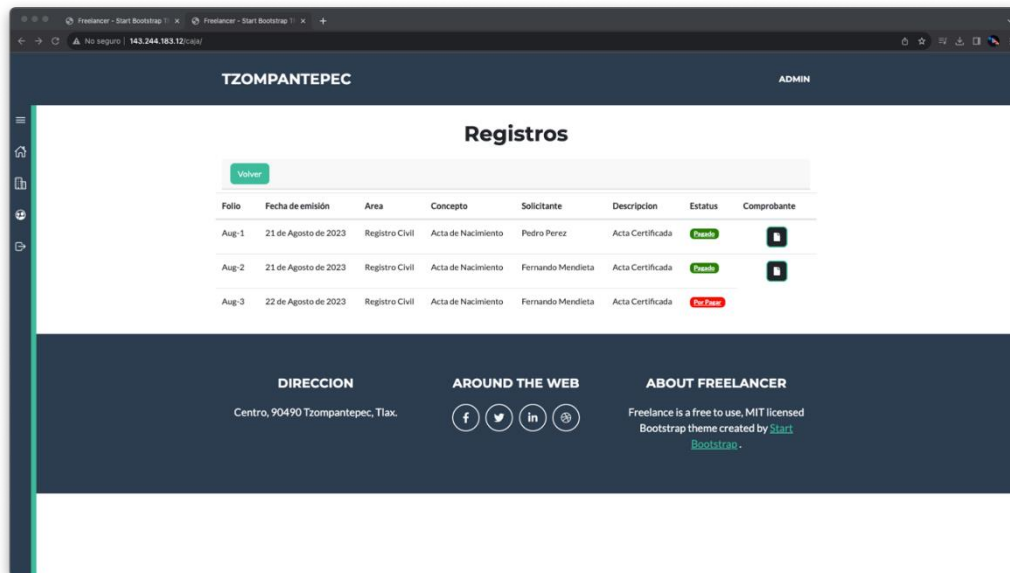


Figura 54. Listado de registros en la sección de caja

En la opción de caja se muestra el listado de las ordenes que se han creado, con la opción de modificar el estatus cuando está ya haya sido pagada, en caso de que si se habilitara el botón de comprobante para ser descargado

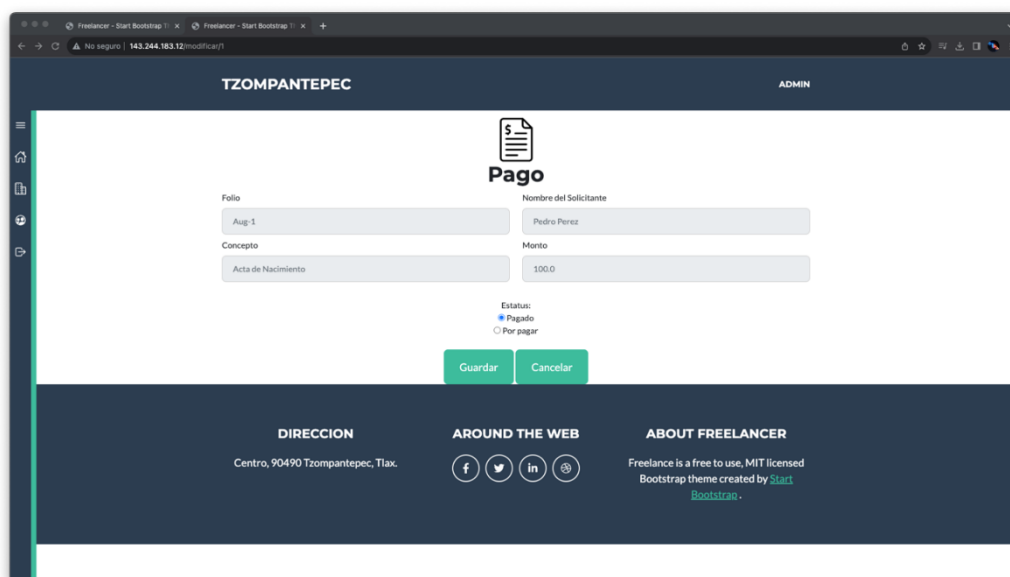


Figura 55. Cambio de estatus a una orden de pago

Se muestra los detalles de la orden que ha sido seleccionada para posteriormente cambiar el estatus de la misma y guardar

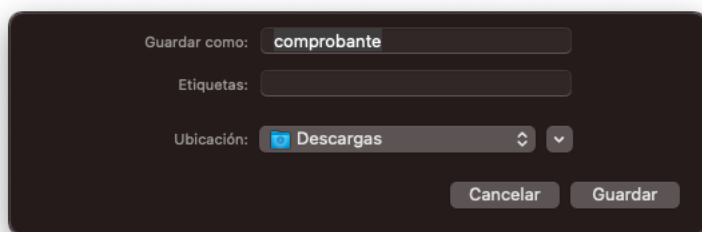


Figura 56. Ubicación de guardado comprobante

Al descargar el comprobante el navegador le va a indicar donde desear guardar el archivo de lo contrario se guardará en la carpeta descargas

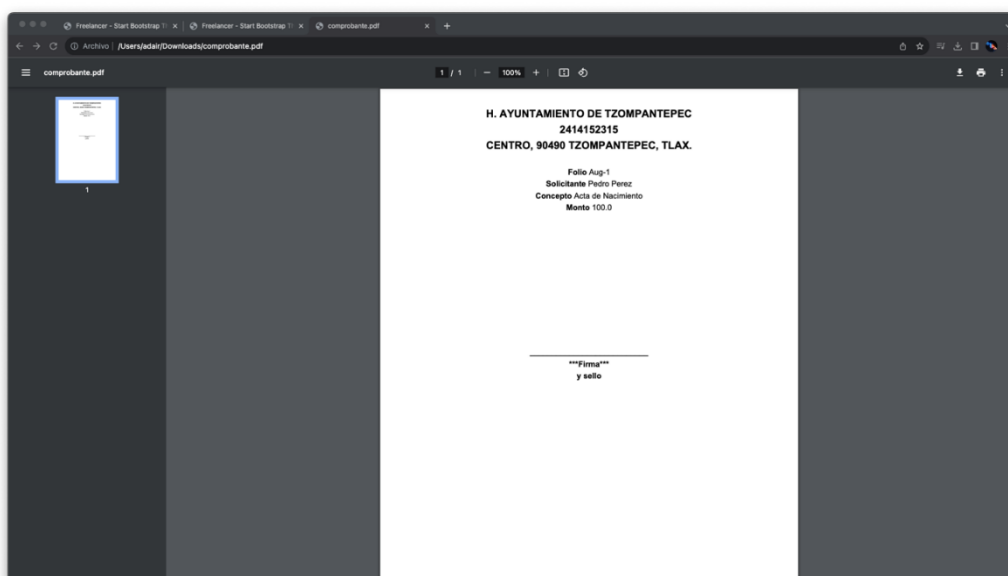


Figura 57. Comprobante generado

Una vez abierto el documento generado se puede imprimir para posteriormente entregarlo al contribuyente

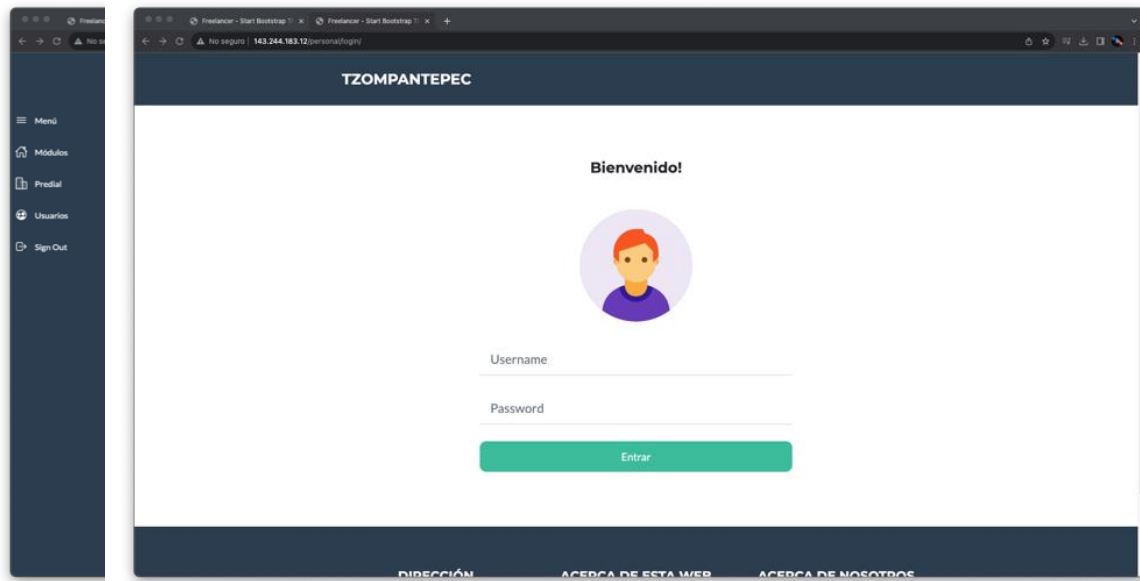


Figura 58. Menú y cerrado de sesión

En el menú se muestra la opción de cerrar sesión, si se elige esta opción se redirigirá al inicio de sesión para poder iniciar con otro usuario o terminar de usar el sistema

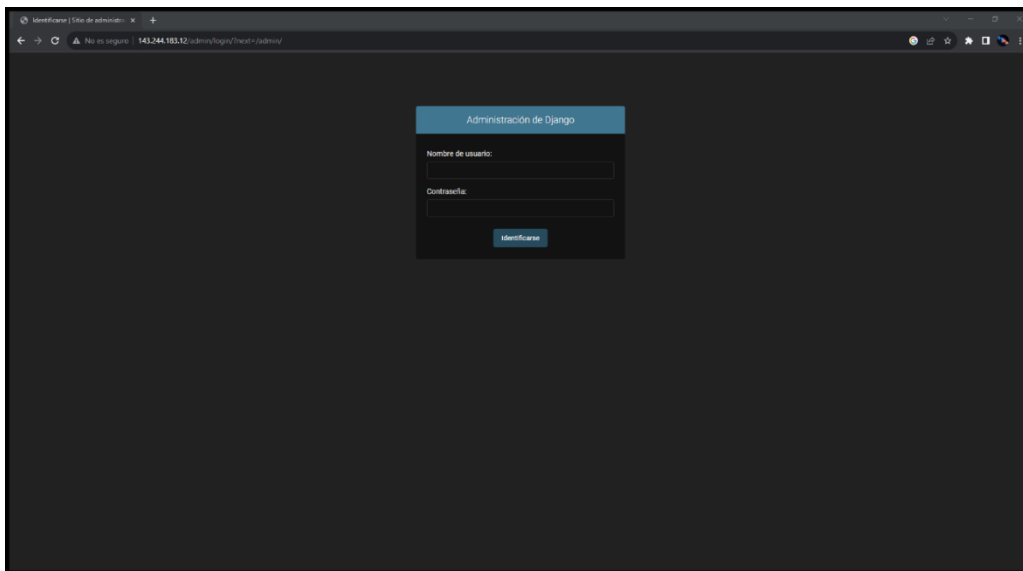


Figura 59. Inicio de sesión administrador

Para iniciar sesión en el módulo del administrador en la url se debe aumentar /admin el cual se mostrará el inicio de sesión para el administrador, aquí solo puede iniciar sesión el usuario que tenga esos privilegios

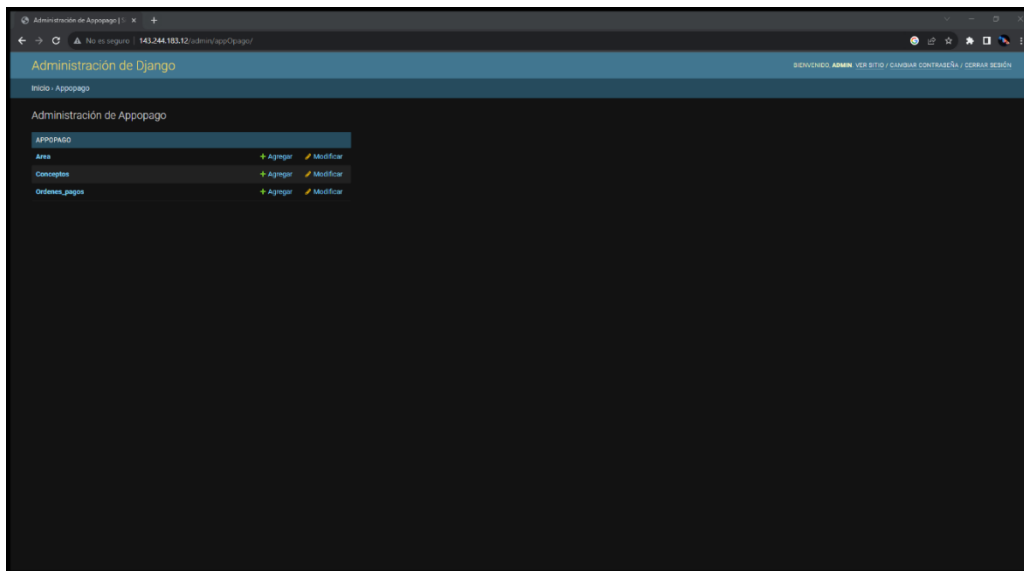


Figura 60. Gestión de modelos a través del administrador

Para el administrador se muestran los modelos que se registraron anteriormente en el código, con la opción de agregar, modificar y eliminar ya que tiene todos los permisos sobre la aplicación

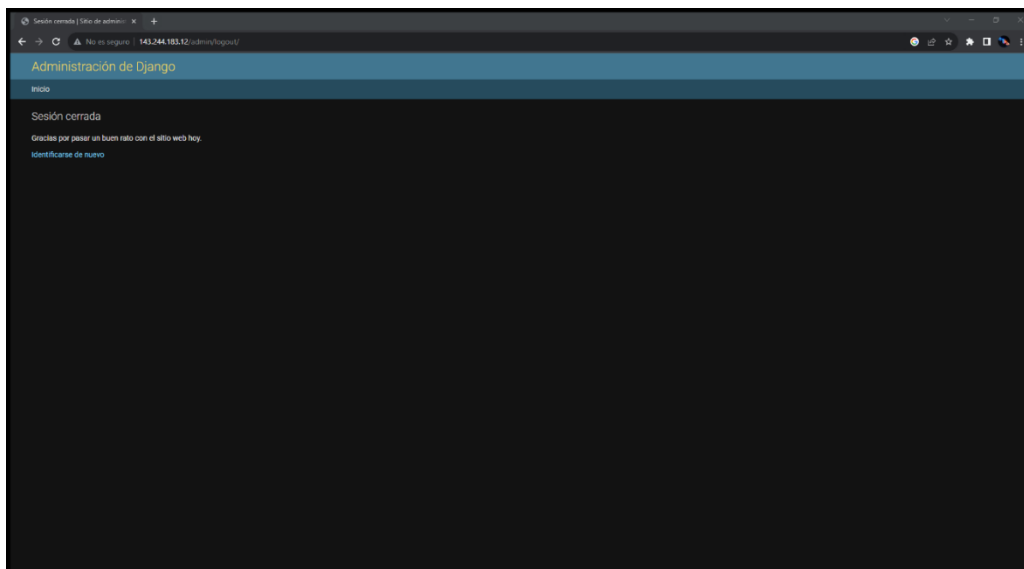


Figura 61. Cierre de sesión administrador

Al terminar de interactuar con la aplicación de las ordenes de pago a través del administrador se puede cerrar sesión la cual se encuentra en la parte superior derecha o bien no cerrar sesión si no ir al sistema

Plan de Pruebas

Prueba Funcional			
Historia de usuario No. 1 Nombre:		Versión de la ejecución	
		Fecha de ejecución	
Modulo del sistema		Loguin	
Descripción de la historia de usuario		Quiero poder iniciar sesión en el sistema utilizando mi nombre de usuario y contraseña para acceder a mi cuenta personalizada	
	Pasos de la historia de usuario		
1.	1. El usuario ingresa el dominio en el navegador 2. El usuario ingresa el nombre de usuario 3. El usuario ingresa la contraseña 4. El usuario da clic en el botón de entrar 5. El usuario visualiza las acciones por hacer		
Paso	Respuesta esperada de la aplicación	Cumple Si/No	Respuesta del sistema
1.	El usuario ve un formulario con información para llenar como lo es el nombre de usuario y la contraseña	Si	Se muestra el formulario con el titulo y el campo para ingresar texto
2.	El usuario da clic en el botón de entrar, si sus credenciales son correctas lo redirige a la pestaña de acciones de lo contrario le indica que alguno de los dos campos es incorrecto y que vuelva a ingresar sus credenciales	Si	El botón inicia la acción ya sea a la pestaña de acciones o que vuelva a ingresar sus credenciales
Resultados de la prueba			
Resultados inesperados o fallos		Juicio	
Sin fallos		Aprobado	
Tiempo estimado		Tiempo real	
Observaciones		Probador	
El modulo cumple con las funciones descritas		Firmas:	

	Nombre: Adair Gustavo Montiel Hernandez
	Verifico
	Fecha:

Prueba Funcional			
Historia de usuario No. 2 Nombre:		Versión de la ejecución	
		Fecha de ejecución	
Modulo del sistema		Registros	
Descripción de la historia de usuario		Quiero tener la capacidad de ver una lista completa de las órdenes de pago	
	Pasos de la historia de usuario		
1.	1. El usuario visualiza la pestaña de acciones 2. El usuario da clic en el botón de acceder 3. El usuario ve una tabla con las órdenes de pago realizadas, con columnas principales 4. El usuario da clic en el botón elegido para exportar la tabla a pdf, Excel etc. 5. El usuario visualiza el archivo descargado en la sección de descargas de su navegador		
Paso	Respuesta esperada de la aplicación	Cumple Si/No	Respuesta del sistema
1.	El usuario ve el titulo de la ventana a la que accedió	Si	El sistema muestra correctamente la venta
2.	El usuario ve los botones de exportar rotulados con el tipo de archivo	Si	Los botones del sistema son funcionales
3.	El usuario visualiza el archivo descargado en la sección de descargas de su navegador	Si	El archivo se descarga correctamente
4.	El usuario visualiza la tabla de las ordenes de pago realizadas	Si	La tabla se muestra correctamente con sus filas y columnas adecuadas
Resultados de la prueba			
Resultados inesperados o fallos		Juicio	

Sin fallos	Aprobado
Tiempo estimado	Tiempo real
Observaciones	Probador
El sistema cumple con las funciones descritas	Firmas: Nombre: Adair Gustavo Montiel Hernandez Verifico Fecha:

Prueba Funcional			
Historia de usuario No. 3 Nombre:		Versión de la ejecución	
		Fecha de ejecución	
Modulo del sistema		Ordenes	
Descripción de la historia de usuario		Necesito realizar órdenes de pago en el sistema	
	Pasos de la historia de usuario		
1.	1. El usuario visualiza la pantalla de acciones 2. El usuario da clic en el botón de acceder al modulo de ordenes 3. El usuario visualiza la ventana de las órdenes de pago 4. El usuario da clic en el botón de nuevo 5. El usuario es dirigido al formulario para llenar los campos de la orden 6. El usuario da clic en el botón de generar orden 7. El usuario es redireccionado a la ventana de ordenes de pago y visualiza la orden que ha creado		
Paso	Respuesta esperada de la aplicación	Cumple Si/No	Respuesta del sistema
1.	El usuario ve la ventana de órdenes de pago con el titulo correspondiente al modulo que abrir	Si	El sistema muestra los botones funcionales, además de la tabla con sus columnas y filas correspondientes
2.	El usuario visualiza en la ventana el botón de nueva orden y una tabla con las	Si	El sistema redirige al siguiente formulario al dar clic en el botón

	ordenes que ha creado		
3.	El usuario visualiza otro formulario donde van a ser llenados los campos solicitados	Si	El sistema muestra correctamente el formulario para ser llenado
4.	El usuario es redirigido a la venta de ordenes al dar clic en el botón de generar la orden	Si	El sistema guarda correctamente el formulario y redirige a la anterior ventana
Resultados de la prueba			
Resultados inesperados o fallos		Juicio	
Sin fallos		Aprobado	
Tiempo estimado		Tiempo real	
Observaciones		Probador	
El sistema y las ventanas cumple con su función descrita		Firmas:	
		Nombre: Adair Gustavo Montiel Hernandez	
		Verifico	
		Fecha:	

Prueba Funcional		
Historia de usuario No. 4 Nombre:	Versión de la ejecución	
	Fecha de ejecución	
Modulo del sistema	Registros	
Descripción de la historia de usuario	Deseo generar informes y estadísticas relacionados con las órdenes de pago	
	Pasos de la historia de usuario	
1.	1. El usuario ubica la tarjeta de registros 2. El usuario da clic en el botón de acceder 3. El usuario visualiza en la ventana el título y posteriormente un botón de volver 4. El usuario visualiza que se muestran 5 botones en el cual puede exportar la tabla a cualquier tipo de archivo que desee 5. El usuario visualiza un cuadro de búsqueda en el que puede introducir	

	cualquier dato para hacer el filtro 6. El usuario visualiza una tabla que donde se muestran las órdenes que se han realizado 7. El usuario visualiza en la ultima columna el estatus de la orden		
Paso	Respuesta esperada de la aplicación	Cumple Si/No	Respuesta del sistema
1.	Al dar clic en el botón de acceder redirige a la nueva ventana de reportes	Si	El sistema carga correctamente la ventana de reportes
2.	El botón de volver regresa al inicio de las ordenes de pago	Si	El sistema cumple con la función del botón que regresa a la venta anterior
3.	El botón de copiar, hace el copiado de la tabla	Si	El sistema cumple con la función del botón, la tabla se puede pegar en un documento de texto
4.	El botón de csv exporta la tabla	Si	El sistema cumple con la función del botón, el archivo se abre correctamente cuando es descargado
5.	El botón de Excel exporta la tabla	Si	El sistema cumple correctamente con la función del botón, el archivo es descargado automáticamente
6.	El botón de pdf exporta la tabla	Si	El sistema cumple con la función del botón, automáticamente se descarga el archivo
7.	El botón de print abre la ventana de impresión del navegador	Si	El sistema cumple con la función del botón, se abre la ventana de impresión con la tabla
8.	El cuadro de búsqueda hace el filtrado del dato que se ingresa	Si	El sistema cumple con la función del cuadro de búsqueda dinámicamente se filtra la tabla al introducir algún dato
9.	El estatus de la orden cambia entre emitida y pagado	Si	El sistema cumple con la función de indicar si la orden ha sido emitida o ha sido pagada
Resultados de la prueba			

Resultados inesperados o fallos	Juicio
Sin fallos	Aprobado
Tiempo estimado	Tiempo real
Observaciones	Probador
El sistema cumple con todas las funciones descritas	Firmas: Nombre: Adair Gustavo Montiel Hernandez Verifico Fecha:

Prueba Funcional			
Historia de usuario No. 5 Nombre:		Versión de la ejecución	
		Fecha de ejecución	
Modulo del sistema		Cualquiera	
Descripción de la historia de usuario		Quiero tener acceso en línea para ver y gestionar las órdenes de pagó	
	Pasos de la historia de usuario		
1.	1. El usuario ingresa en la URL de su navegador el dominio del sistema 2. El usuario visualiza el inicio de sesión del sistema		
Paso	Respuesta esperada de la aplicación	Cumple Si/No	Respuesta del sistema
1.	El usuario visualiza la ventana en la que debe iniciar sesión	Si	El sistema carga correctamente
Resultados de la prueba			
Resultados inesperados o fallos		Juicio	
Sin fallos		Aprobado	
Tiempo estimado		Tiempo real	
Observaciones		Probador	
El sistema carga correctamente al abrirlo desde cualquier navegador		Firmas:	
		Nombre: Adair Gustavo Montiel Hernandez	

	Verifico
	Fecha:

Prueba Funcional			
Historia de usuario No. 6 Nombre:		Versión de la ejecución	
		Fecha de ejecución	
Modulo del sistema		Conceptos	
Descripción de la historia de usuario		Se debe implementar una validación exhaustiva de los montos ingresados en las órdenes de pago	
	Pasos de la historia de usuario		
1.	1. El usuario elije la tarjeta de conceptos 2. El usuario elige acceder dando clic en el botón de acceder 3. El usuario da clic en el botón nuevo 4. El usuario visualiza un formulario donde ingresa el nombre, el monto. Y el área a la que pertenece el concepto 5. El usuario visualiza una ventana de la gestión de conceptos, los puede modificar o eliminar		
Paso	Respuesta esperada de la aplicación	Cumple Si/No	Respuesta del sistema
1.	El usuario es redirigido a la ventana de gestión de conceptos al dar clic en el botón de acceder	Si	El sistema carga correctamente la ventana de la gestión de conceptos
2.	El usuario visualiza la ventana de crear concepto al dar clic en el botón de nuevo	Si	El sistema redirecciona correctamente al formulario
3.	El usuario puede exportar la tabla a un tipo de archivo en especifico	Si	El sistema cumple con la función de cada botón al exportar la tabla
4.	El usuario modifica un concepto ya creado dando clic en el botón de modificar	Si	El sistema funciona correctamente al modificar un concepto en cualquier campo
5.	El usuario elimina un concepto dando clic en el icono del bote	Si	El sistema elimina correctamente el concepto

	de basura		seleccionado
Resultados de la prueba			
Resultados inesperados o fallos		Juicio	
No hay ningún fallo		Aprobado	
Tiempo estimado		Tiempo real	
Observaciones		Probador	
El sistema cumple con todas las funciones		Firmas:	
		Nombre: Adair Gustavo Montiel Hernandez	
		Verifico	
		Fecha:	

Prueba Funcional			
Historia de usuario No. 7 Nombre:		Versión de la ejecución	
		Fecha de ejecución	
Modulo del sistema		Se debe registrar las actividades de acceso	
Descripción de la historia de usuario			
	Pasos de la historia de usuario		
1.	1. El usuario modifica la URL del sistema 2. El usuario debe aumentar un /admin a la URL del sistema 3. El usuario inicia sesión con sus credenciales de administrador 4. El usuario se dirige a la opción de usuarios 5. El usuario elije un usuario 6. El usuario ve el historial de acceso		
Paso	Respuesta esperada de la aplicación	Cumple Si/No	Respuesta del sistema
1.	El usuario visualiza el modulo de administrador		El sistema carga correctamente el inicio de sesión para el administrador
2.	El usuario inicia sesión correctamente en el administrador		El sistema redirige al inicio del administrador al introducir correctamente las credenciales de acceso

3.	El usuario visualiza todos los módulos existentes en el sistema		El sistema carga correctamente el administrador
4.	El usuario visualiza la lista de usuarios existentes		El sistema carga la lista de usuarios existentes
5.	El usuario visualiza las actividades de acceso que ha tenido el usuario		El sistema carga correctamente el historial de actividad del usuario seleccionado
Resultados de la prueba			
Resultados inesperados o fallos		Juicio	
Sin ningún fallo		Aprobado	
Tiempo estimado		Tiempo real	
Observaciones		Probador	
El sistema del administrador cumple con las funciones descritas		Firmas: Nombre: Adair Gustavo Montiel Hernandez Verifico Fecha:	

Evaluación de la usabilidad

Formato de aceptación del usuario				
Nombre del módulo:	Loguin			
Pregunta	Si	No	En ocasiones	Observaciones
¿El vocabulario empleado es entendible y se presenta en un solo idioma?				
¿Las interfaces son intuitivas y amigables?				
¿Se identifican los elementos del sistema, como botones, inputs, menús, etc.?				
¿Aprender a usar este sistema es fácil, sin recibir				

ninguna capacitación?				
¿Se muestra información sobre las acciones que realiza el software, notificaciones, alertas, etc.?				
¿Los mensajes del sistema son entendibles?				
¿El sistema notifica al usuario en caso de un caso de uso que no es el esperado?				
¿Permite una forma de navegación simple e intuitiva?				
¿Hay información visual de que es lo que hace cada interfaz y que puede hacer en cada módulo?				
Total	Valor de cada si = 2	Valor de cada no = 0	Valor de en ocasiones = 1	

Formato de aceptación del usuario				
Nombre del módulo:	Áreas del municipio			
Pregunta	Si	No	En ocasiones	Observaciones
¿El vocabulario empleado es entendible y se presenta en un solo idioma?				
¿Las interfaces son intuitivas y amigables?				
¿Se identifican los elementos del sistema, como botones, inputs, menús, etc.?				
¿Aprender a usar este sistema es fácil, sin recibir ninguna capacitación?				
¿Se muestra información sobre las acciones que realiza el software, notificaciones, alertas, etc.?				

¿Los mensajes del sistema son entendibles?				
¿El sistema notifica al usuario en caso de un caso de uso que no es el esperado?				
¿Permite una forma de navegación simple e intuitiva?				
¿Hay información visual de que es lo que hace cada interfaz y que puede hacer en cada módulo?				
Total	Valor de cada si = 2	Valor de cada no = 0	Valor de en ocasiones = 1	

Formato de aceptación del usuario				
Nombre del módulo:	Ordenes de pago			
Pregunta	Si	No	En ocasiones	Observaciones
¿El vocabulario empleado es entendible y se presenta en un solo idioma?				
¿Las interfaces son intuitivas y amigables?				
¿Se identifican los elementos del sistema, como botones, inputs, menús, etc.?				
¿Aprender a usar este sistema es fácil, sin recibir ninguna capacitación?				
¿Se muestra información sobre las acciones que realiza el software, notificaciones, alertas, etc.?				
¿Los mensajes del sistema son entendibles?				
¿El sistema notifica al usuario en caso de un caso de uso que no es el esperado?				

¿Permite una forma de navegación simple e intuitiva?				
¿Hay información visual de que es lo que hace cada interfaz y que puede hacer en cada módulo?				
Total	Valor de cada si = 2	Valor de cada no = 0	Valor de en ocasiones = 1	

Formato de aceptación del usuario				
Nombre del módulo:	Registros / Reportes			
Pregunta	Si	No	En ocasiones	Observaciones
¿El vocabulario empleado es entendible y se presenta en un solo idioma?				
¿Las interfaces son intuitivas y amigables?				
¿Se identifican los elementos del sistema, como botones, inputs, menús, etc.?				
¿Aprender a usar este sistema es fácil, sin recibir ninguna capacitación?				
¿Se muestra información sobre las acciones que realiza el software, notificaciones, alertas, etc.?				
¿Los mensajes del sistema son entendibles?				
¿El sistema notifica al usuario en caso de un caso de uso que no es el esperado?				
¿Permite una forma de navegación simple e intuitiva?				
¿Hay información visual de que es lo que hace cada interfaz y que puede hacer en				

cada módulo?				
Total	Valor de cada si = 2	Valor de cada no = 0	Valor de en ocasiones = 1	

Formato de aceptación del usuario				
Nombre del módulo:	Gestión de conceptos			
Pregunta	Si	No	En ocasiones	Observaciones
¿El vocabulario empleado es entendible y se presenta en un solo idioma?				
¿Las interfaces son intuitivas y amigables?				
¿Se identifican los elementos del sistema, como botones, inputs, menús, etc.?				
¿Aprender a usar este sistema es fácil, sin recibir ninguna capacitación?				
¿Se muestra información sobre las acciones que realiza el software, notificaciones, alertas, etc.?				
¿Los mensajes del sistema son entendibles?				
¿El sistema notifica al usuario en caso de un caso de uso que no es el esperado?				
¿Permite una forma de navegación simple e intuitiva?				
¿Hay información visual de que es lo que hace cada interfaz y que puede hacer en cada módulo?				
Total	Valor de cada si = 2	Valor de cada no = 0	Valor de en ocasiones = 1	

Formato de aceptación del usuario				
Nombre del módulo:	Caja			
Pregunta	Si	No	En ocasiones	Observaciones
¿El vocabulario empleado es entendible y se presenta en un solo idioma?				
¿Las interfaces son intuitivas y amigables?				
¿Se identifican los elementos del sistema, como botones, inputs, menús, etc.?				
¿Aprender a usar este sistema es fácil, sin recibir ninguna capacitación?				
¿Se muestra información sobre las acciones que realiza el software, notificaciones, alertas, etc.?				
¿Los mensajes del sistema son entendibles?				
¿El sistema notifica al usuario en caso de un caso de uso que no es el esperado?				
¿Permite una forma de navegación simple e intuitiva?				
¿Hay información visual de que es lo que hace cada interfaz y que puede hacer en cada módulo?				
Total	Valor de cada si = 2	Valor de cada no = 0	Valor de en ocasiones = 1	

Conclusiones

La implementación del modulo de ordenes de pago en el sistema integral de cobros municipales representa un avance en la eficiencia de la administración del municipio. A lo largo de este proyecto he explorado lo que es la importancia de la integración además de que representa desafíos, pero al mismo tiempo implica beneficios.

Durante el desarrollo de este proyecto he considerado los requerimientos tanto funcionales como no funcionales, así mismo abordando los temas que considero que son de importancia como lo es la seguridad, la experiencia del usuario y la integración correcta con los demás módulos.

El uso de framework de Django me ha permitido crear una plataforma robusta y escalable que satisface las necesidades del municipio de Tzompantepec Apizaco.

Este proyecto tiene las bases para futuras mejoras y expansiones del sistema, así mismo demuestra la integración de tecnologías pudiendo mejorar la calidad de los servicios públicos y fortalecer la relación entre la administración y la comunidad.

Competencias desarrolladas

- ✓ Desarrollo de software
Habilidades en el desarrollo de software utilizando el framework de Django, lo que permitió diseñar y crear el modulo de ordenes de pago de manera efectiva y eficiente
- ✓ Análisis de requerimientos
Identificar y analizar los requerimientos del sistema a partir de las necesidades y expectativas del usuario, lo que permite traducir esas necesidades en características funcionales
- ✓ Control de versiones en Git
Conocimientos y habilidades en el uso de Git, para la gestión de manera efectiva las diferentes versiones del código fuente y así mismo colaborar con otros miembros del equipo y mantener un historial claro de los cambios realizados en el proyecto
- ✓ Conexión a la Base de Datos PostgreSQL con Django
Configuración y gestión de la conexión entre el framework Django y la base de datos PostgreSQL.
- ✓ Diseño de interfaz de usuario
Diseñar la interfaz de usuario intuitiva y amigable para los usuarios, mejorando su experiencia y facilitando la interacción con el sistema
- ✓ Gestión de proyectos
Habilidades en la planificación, organización y seguimiento de proyectos asegurando que las etapas del desarrollo se completen de manera efectiva y dentro de los plazos establecidos.
- ✓ Seguridad
Implementación de medidas de seguridad en el sistema, protegiendo la información sensible previniendo posibles ataques
- ✓ Integración de módulos
Lograr integrar el modulo de ordenes de pago con el sistema integral de cobros municipales, demostración de la capacidad para trabajar con sistemas preexistentes
- ✓ Resolución de problemas
Desafíos técnicos y de diseño durante el proyecto y demostración de la habilidad para identificar soluciones efectivas y superar el obstáculo
- ✓ Gestión de datos
Trabajo con bases de datos para almacenar y gestionar información relacionada con las ordenes de pago
- ✓ Colaboración

Colaboración con diferentes compañeros demostrando la capacidad de trabajar en equipos para alcanzar objetivos comunes

Fuentes de información

- * Corrales, J. A. (2020, junio 28). Experiencia del usuario: la guía completa para complacer a tu audiencia digital en el 2020. Rock Content - ES; Rock Content. <https://rockcontent.com/es/blog/experiencia-del-usuario/>
- * Graván, A. D. (2018, noviembre 29). Manifiesto “agile”, ¿qué es? BBVA. <https://www.bbva.com/es/innovacion/agile-manifiesto-que-es/>
- * Introducción a Django. (s/f). Mozilla.org. Recuperado el 20 de agosto de 2023, de <https://developer.mozilla.org/es/docs/Learn/Server-side/Django/Introduction>
- * Que es el patrón MTV (Model Template View). (s/f). EspiFreelancer. Recuperado el 20 de agosto de 2023, de <https://espifreelancer.com/mtv-django.html>
- * ¿Qué es la metodología ágil? (s/f). Redhat.com. Recuperado el 20 de agosto de 2023, de <https://www.redhat.com/es/devops/what-is-agile-methodology>
- * ¿Qué es PostgreSQL? (s/f). Ibm.com. Recuperado el 20 de agosto de 2023, de <https://www.ibm.com/mx-es/topics/postgresql>
- * Qué es SCRUM. (2008, agosto 4). Proyectos Ágiles. <https://proyectosagiles.org/que-es-scrum/>
- * Raeburn, A. (2022, noviembre 28). ¿Qué es la programación extrema (XP)? [2022] •. Asana. <https://asana.com/es/resources/extreme-programming-xp>
- * (S/f-a). Amazon.com. Recuperado el 20 de agosto de 2023, de <https://aws.amazon.com/es/what-is/python/#:~:text=Python%20es%20un%20lenguaje%20interpretado,en%20el%20c%C3%B3digo%20con%20rapidez.>
- * (S/f-b). Amazon.com. Recuperado el 20 de agosto de 2023, de <https://aws.amazon.com/es/what-is/django/#:~:text=Django%20es%20un%20software%20que,y%20la%20administraci%C3%B3n%20de%20cookies.>
- * (S/f-c). Udemy.com. Recuperado el 20 de agosto de 2023, de <https://www.udemy.com/course/domina-el-orm-de-django/>

Anexos

Carta de Aceptación

 ESTADOS UNIDOS MEXICANOS
UNIDOS POR UN DESARROLLO

 PRESIDENCIA MUNICIPAL **TZOMPANTEPEC**
H. AYUNTAMIENTO 2021-2024

Apizaco, Tlax., a 7 de febrero de 2023

ASUNTO: CARTA DE ACEPTACIÓN
DE RESIDENCIA PROFESIONAL

MTRA. NORMA ANGELICA AGUIÑAGA SÁNCHEZ
JEFA DEL DEPARTAMENTO DE GESTIÓN TECNOLÓGICA
Y VINCULACIÓN
P R E S E N T E

Por este conducto reciba un cordial saludo, y a la vez me permito informar a usted que el **C. Adair Gustavo Montiel Hernández**, con Número de Control **18370981** alumno de la carrera de Ingeniería en Tecnologías de la Información y Comunicaciones, ha sido aceptado para realizar su Residencia Profesional; en el periodo comprendido **del 7 de febrero a 23 de junio de 2023** bajo la asesoría del Lic. Paulino Ramos González, Secretario de este Ayuntamiento.

Sin mas por el momento, me despido de usted agradeciendo la atención prestada.

ATENTAMENTE


ING. MANUEL RAMOS MONTIEL
PRESIDENTE MUNICIPAL CONSTITUCIONAL
DE TZOMPANTEPEC

 ESTADOS UNIDOS MEXICANOS
PRESIDENCIA MUNICIPAL
TZOMPANTEPEC
TLAX. H. AYUNTAMIENTO 2021-2024

c.c.p. Archivo

Municipio de Tzompantepec, Tlax. Zaragoza No. 1, Col. Centro San Salvador Tzomp. C.P. 90490.
Tel: 241 41 52315

Carta de Liberación



UNIDOS POR UN DESARROLLO



PRESIDENCIA
MUNICIPAL

TZOMPANTEPEC
H. AYUNTAMIENTO
2021-2024

Tzompantepec, Tlax., 15/Agosto/2023

ASUNTO: **LIBERACIÓN DE RESIDENCIA
PROFESIONAL.**

DRA. YESICA IMELDA SAAVEDRA BENITEZ

DIRECTORA DEL TecNM / INSTITUTO TECNOLÓGICO DE APIZACO

PRESENTE

Por este medio me permito informar a usted que el H. Ayuntamiento Municipal Constitucional de Tzompantepec, **LIBERA** la realización de **RESIDENCIA PROFESIONAL** del **C. ADAIR GUSTAVO MONTIEL HERNÁNDEZ**, alumno de la carrera de **INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIONES** del Instituto que usted dignamente representa, con **No. de Control 18370981**, quien durante su estancia desarrolló el proyecto denominado **"DESARROLLO E INTEGRACIÓN DEL MÓDULO DE ORDENES DE PAGO AL SISTEMA INTEGRAL DE COBROS MUNICIPALES"** que de acuerdo con lo estipulado en la carta de presentación cubrió un total de 500 horas, durante el periodo comprendido del **07 de Febrero al 23 de Junio de 2023**.

Sin otro particular por el momento, reciba un cordial saludo.

ATENTAMENTE


ING. MANUEL RAMOS MONTIEL
**PRESIDENTE MUNICIPAL CONSTITUCIONAL
DE TZOMPANTEPEC**



PRESIDENCIA
MUNICIPAL
TZOMPANTEPEC
TLAX. H. AYUNTAMIENTO
2021-2024

c.c.p. Archivo

Municipio de Tzompantepec, Tlax. Zaragoza No. 1, Col. Centro San Salvador Tzomp. C.P. 90490.
Tel: 241 41 52315

Carta de No inconveniencia



UNIDOS POR UN DESARROLLO



PRESIDENCIA
MUNICIPAL

TZOMPANTEPEC
H. AYUNTAMIENTO
2021-2024

Apizaco, Tlaxcala, 31/Agosto/2023

Asunto: CARTA DE NO INCONVENIENCIA

DRA. YESICA IMELDA SAAVEDRA BENITEZ

DIRECTORA DEL TecNM / INSTITUTO TECNOLÓGICO DE APIZACO

PRESENTE

Referente a la atenta solicitud del **C. ADAIR GUSTAVO MONTIEL HERNANDEZ**, con número de control **18370981**, estudiante de la carrera de **INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIONES**, de la Institución educativa a su digno cargo, comisionado en este H. Ayuntamiento, donde realizó cabalmente su "RESIDENCIA PROFESIONAL", durante el periodo del **07 de Febrero al 23 de Junio de 2023**, por medio del presente informo a usted que **No Existe Inconveniente** alguno para que el interesado utilice con fines de titulación el proyecto denominado:

"DESARROLLO E INTEGRACIÓN DEL MÓDULO DE ORDENES DE PAGO AL SISTEMA INTEGRAL DE COBROS MUNICIPALES"

ATENTAMENTE

ING. MANUEL RAMOS MONTIEL

**PRESIDENTE MUNICIPAL CONSTITUCIONAL
DE TZOMPANTEPEC**



c.c.p. Archivo

Municipio de Tzompantepec, Tlax. Zaragoza No. 1, Col. Centro San Salvador Tzomp. C.P. 90490.
Tel: 241 41 52315