UNIVERSIDAD DE EL SALVADOR

FACULTAD MULTIDISCIPLINARIA DE OCCIDENTE

DEPARTAMENTO DE INGENIERÍA Y ARQUITECTURA

TRABAJO DE GRADO

DISEÑO Y DESARROLLO DE UN SISTEMA INFORMÁTICO PARA LA ADMINISTRACIÓN Y GESTIÓN DE COBRANZAS DEL SERVICIO DE AGUA POTABLE PARA LA UNIDAD DE ADMINISTRACIÓN TRIBUTARIA DE LA ALCALDÍA MUNICIPAL DE CHALCHUAPA

PARA OPTAR AL GRADO DE

INGENIERO DE SISTEMAS INFORMÁTICOS

PRESENTADO POR

VÍCTOR ISMAEL MARTÍNEZ RODRÍGUEZ

ERICK ADIEL TRIGUEROS JEREZ

GUSTAVO ADOLFO VELÁSQUEZ PLEITEZ

DOCENTE ASESOR

INGENIERO RICARDO MISAEL AYALA MOLINA

AGOSTO, 2019

SANTA ANA, EL SALVADOR, CENTRO AMÉRICA

UNIVERSIDAD DE EL SALVADOR

AUTORIDADES

M.Sc. ROGER ARMANDO ARIAS ALVARADO

RECTOR

DR. MANUEL DE JESÚS JOYA ABREGO

VICERRECTOR ACADÉMICO

ING. NELSON BERNABÉ GRANADOS ALVARADO

VICERRECTOR ADMINISTRATIVO

LICDO. CRISTOBAL HERNÁN RÍOS BENÍTEZ

SECRETARIO GENERAL

M.Sc. CLAUDIA MARÍA MELGAR DE ZAMBRANA

DEFENSORA DE LOS DERECHOS UNIVERSITARIOS

LICDO. RAFAEL HUMBERTO PEÑA MARÍN

FISCAL GENERAL

FACULTAD MULTIDISCIPLINARIA DE OCCIDENTE

AUTORIDADES

DR. RAÚL ERNESTO AZCÚNAGA LÓPEZ

DECANO

M.Ed. ROBERTO CARLOS SIGÜENZA CAMPOS

VICEDECANO

M.Sc. DAVID ALFONSO MATA ALDANA

SECRETARIO

ING. DOUGLAS GARCÍA RODEZNO

JEFE DEL DEPARTAMENTO DE INGENIERÍA Y ARQUITECTURA

Agradezco a Dios todo poderoso, porque me ha dotado de sabiduría absoluta, y en este proyecto me guio e iluminó con las ideas exactas durante toda la etapa de desarrollo del proyecto.

A mi madre, por brindarme todo su apoyo y recursos durante todos los años de estudio hasta finalizar la carrera.

A mi familia, por apoyarme en los momentos de decadencia brindándome las palabras de ánimo, fortaleza y hacerme entrar en razón que tengo la capacidad idónea para seguir en esta profesión.

A los administradores de la Alcaldía de Chalchuapa, por haber aprobado y colaborado durante el desarrollo del proyecto, y por confiar en nuestras capacidades para solventar el problema que ha tenido la institución en la mala gestión de cobros del servicio de agua potable.

A mis compañeros, porque apoyaron y confiaron plenamente en todo momento en la idea para realizar el desarrollo del proyecto.

Tutor de tesis, gracias por las sugerencias y motivaciones que nos brindó durante el proyecto.

Víctor Ismael Martínez Rodríguez

Primeramente, dar gracias a Dios todo poderoso que me ha brindado protección todos estos años y me ha dado puesto las personas y herramientas necesarias para poder culminar mis estudios, gracias a él que me ha dado las fuerzas para seguir aun cuando me he sentido derrotado en este largo camino, que ha tenido noches de desvelo, alegrías, decepciones y sobre todo lleno de aprendizaje para así forjarme como un buen profesional y servir como un buen elemento para dar soluciones a los problemas tecnológicos que se puedan dar en el ámbito que estamos inmersos.

También dar gracias a mi familia que me ha apoyado en todo momento, estando siempre en mis alegrías y en mis momentos de estrés, especialmente a mi tía Rosario de Hernández y mi tío Jaime Jerez que me han brindado consejos, ánimos y los recursos para poder culminar mis carrera, a mi mamá Sandra Jerez, tía Aydee Jerez, tío Santos Jerez y abuela Francisca Portillo que me han cuidado, educado y siempre han estado a mi lado incondicionalmente durante todas las etapas de mi vida, a mi hermana Vanessa que ha crecido junto a mí, y me ha brindado consejos. En fin, infinitas gracias a toda mi familia que de una forma u otra han contribuido a mi logro.

Gracias a mis compañeros de estudio, con los cuales nos hemos apoyado mutuamente y hemos tenido buenas relaciones interpersonales, siempre teniendo buenos momentos en el ámbito estudiantil y a través de diferentes ámbitos de entretenimiento. Infinitas gracias a mi compañero y amigo Carlos Alexander Cárcamo que ha estado apoyándome incondicionalmente a través de mi etapa universitaria, también infinitas gracias a su madre Aida González que siempre mostro aprecio y apoyo a mi persona.

Gracias a todos mis docentes que brindaron de sus conocimientos para contribuir a mi aprendizaje.

Gracias a mis amistades que han estado pendiente de mí y apoyándome para ser mejor persona, y que siempre han creído en mis capacidades. Gracias por su apoyo y cariño.

Y que más decir, gracias a mis compañeros de tesis Gustavo Velásquez con el cual hemos sido buenos compañeros durante el transcurso de nuestra carrera y que ha demostrado excelentes capacidades y Víctor Martínez que hemos sido buenos compañeros de poco tiempo, pero que en el transcurso del trabajo de grado demostró ser un buen compañero con excelentes habilidades; gracias a ellos por tener esa paciencia y grado de comprensión para trabajar en equipo, y por tener un alto grado de interés y excelentes habilidades comunicativas en el lapso del diseño y desarrollo del sistema.

Erick Adiel Trigueros Jerez

Primeramente, quiero agradecer a Dios por darme la salud, la protección, la fuerza y la sabiduría para seguir siempre adelante en los mementos más difíciles de la carrera; ya que sin su ayuda, todo esto no podría haber sido posible.

También agradecer a mi familia por su apoyo incondicional a lo largo de mi carrera. A mis padres, Patricia Pleitez y Rodolfo Velásquez por apoyarme en todo momento y por darme los recursos para culminar mi carrera. A mis abuelos, Carmen Pleitez y Carlos Pleitez por todos sus consejos, por todos sus sacrificios y por luchar y estar siempre a mi lado desde el primer día. A mis tías Marta Pleitez y Daisy Pleitez por tenderme siempre la mano en los momentos en que tenía necesidad. Y a mis demás familiares que de una u otra forma me dieron su apoyo.

Gracias a los amigos y compañeros que conocí a lo largo de todos estos años. Un especial agradecimiento a mis amigos Cecilia Romero, Douglas Tobar y Miguel Pacheco, por regalarme su amistad y brindarme su apoyo a lo largo de mi carrera universitaria y de mi vida personal. También quisiera agradecer a mis compañeros de tesis Erick Trigueros y Víctor Martínez, por poner todo su esfuerzo para la culminación de este proyecto; gracias por toda la paciencia y dedicación mostrada. Y a los demás compañeros que conocí a lo largo de mi carrera, muchas gracias.

Gracias a todos los docentes que brindaron de sus conocimientos en las distintas materias de la carrera.

A nuestro docente asesor Ing. Ricardo Ayala, por guiarnos y apoyarnos a lo largo de este trabajo.

Gracias a la Alcaldía Municipal de Chalchuapa, por abrirnos las puertas y permitirnos realizar nuestro proyecto.

Gustavo Adolfo Velásquez Pleitez

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

En la actualidad, los sistemas informáticos se han vuelto imprescindibles en muchas áreas de la sociedad y están a la vanguardia en el manejo y automatización de tareas relacionadas con cualquier tipo de rubro en el ámbito público y privado. Es de conocimiento que para cualquier persona que está detrás de un computador sin un sistema que le resuelva sus necesidades de manera efectiva y eficaz, tiende a adoptar estrés y generarlo en las personas a las que se les brinda atención en una determinada área.

En la búsqueda de mejorar y suplir las necesidades de automatizar procesos en instituciones públicas, se descubre que en la Alcaldía Municipal de Chalchuapa se tiene la necesidad de un sistema informático que automatice y agilice las tareas de la Unidad de Administración Tributaria Municipal (UATM) para la administración y gestión de cobranzas del servicio de agua potable.

Haciendo un estudio acerca del problema se ha llegado a la conclusión de que el uso de un sistema es de gran necesidad para esta área, dado que los procesos se manejan de forma manual y al mismo tiempo usando software ofimático haciendo engorroso el trabajo del operador de dicha área y limitando la posibilidad de hacer un estudio o auditoría de estos procesos de forma eficaz.

Al mismo tiempo, se busca agilizar los procesos, debido que los contribuyentes a la hora de cumplir con sus obligaciones tributarias de dicho servicio hacen una serie de pasos extras, de manera que los tiempos para atender a cada contribuyente se extienden, sabiendo

que en este tipo de procesos el menor tiempo de atención y el estrés mínimo en el contribuyente son de gran importancia.

En el presente documento se presenta a detalle la problemática que enfrenta la municipalidad en la administración y gestión de cobranzas del servicio de agua potable para la Unidad de Administración Tributaria Municipal.

Los antecedentes, en donde se incluye un breve desarrollo histórico sobre la forma en que se realizan los procesos y las herramientas con las que cuentan, justificando porqué es de suma importancia para la unidad municipal, logrando así garantizar una solución efectiva para la agilización de todos los procesos en la unidad.

Además, se detalla el proceso de diseño y creación de la estructura del sistema, el diseño de la base de datos, diagramas de clase, diagramas de casos de uso y se establece el diseño de las interfaces.

También tiene como objetivo presentar las técnicas y tecnologías elegidas para el desarrollo de la solución informática, así como también el diseño y la implementación de las distintas pruebas del sistema que permitan observar la funcionalidad del este.

OBJETIVOS

Objetivo General

- Diseñar y desarrollar un sistema informático fiable que automatice la gestión de pagos de agua en la Unidad de Administración Tributaria de la Alcaldía Municipal de Chalchuapa.

Objetivos Específicos

- Reducir el tiempo de respuesta en la obtención de información a las solicitudes de la población.
- Diseñar una base de datos que se adapte a los cambios futuros en el software y garantice la integridad y persistencia de la información.
- Implementar las buenas prácticas de programación para sostener el mantenimiento y evolución del software.
- Realizar toda planificación y cambios en el proyecto con el cliente para prevenir los fracasos en el software.
- Utilizar un framework para automatizar tareas comunes y organizar mejor el proyecto.
- Diseñar el sistema informático para que posea una vista elegante y sencilla de usar para que el usuario se sienta cómodo realizando sus labores.

CAPÍTULO I

GENERALIDADES DEL PROYECTO

CAPITULO I GENERALIDADES DEL PROYECTO

1.1 MUNICIPIO DE CHALCHUAPA

1.1.1 Organización territorial

Es una ciudad del distrito de Chalchuapa del departamento de Santa Ana, en la zona occidental del país, a 13 km al oeste de Santa Ana y a 78 km de San Salvador. Tiene una extensión territorial de 165,76 km^2; aproximadamente.

Limita al norte con el departamento de Jutiapa, Guatemala; al este con los municipios de Candelaria de la Frontera, El Porvenir, San Sebastián Salitrillo y Santa Ana; al sur con Nahuizalco y Juayúa (ambos del departamento de Sonsonate); y al oeste con San Lorenzo, Atiquizaya y el Refugio (todos pertenecientes al departamento de Ahuachapán).Para su administración se divide en 21 cantones y varios caseríos.

1.1.2 Generalidades del municipio

Chalchuapa que en idioma "NÁHUATL" significa "RÍOS DE JADEÍTAS", fue el más notable emporium de la civilización de los pokomames, pueblos de la familia Maya-Quiche o Máyense y constituye sin duda alguna la zona arqueológica más notable del país, con cinco centros ceremoniales: (Tazumal, Pampe, El Trapiche, Casa Blanca y las Victorias y además, La Laguna Cuzcachapa) estos vestigios precolombinos se encuentran diseminados en un área de 6 km. Cuadrados[1].

[1]Chalchuapa un lugar para visitar (2019, Octubre)
Recuperado de: http://culturachalchuapaneca.blogspot.com/2009/10/historia-de-chalchuapa.html

Es un importantísimo sitio arqueológico. Lugares como Tazumal, Casa blanca, El Trapiche, Las Victorias, Laguna Seca y Laguna Cuzcachapa, entre otros, constituyen las mejores muestras de presencia prehispánica en el territorio. Otro dato curioso y significativo es que la ciudad atravesó todos los períodos arqueológicos, convirtiéndose también en la prueba más antigua de ocupación humana en El Salvador.

En los alrededores afloran a la superficie los vestigios materiales de estos antiguos grupos que habitaron la zona; entre estos se han encontrado piezas de cerámica, grabados, piedras talladas, jade y obsidiana[2].

1.2 MARCO HISTÓRICO ALCALDÍA MUNICIPAL DE CHALCHUAPA

1.2.1 Gobierno Municipal

La alcaldía en un Gobierno Local encargado de la administración, funcionamiento y ejercicio de las atribuciones de los municipios cuya autoridad máxima es el Consejo Municipal. Es también la entidad encargada de velar por el desarrollo social, económico y cultural del municipio.

1.2.2 Localización

La Alcaldía Municipal de Chalchuapa se encuentra ubicada en 2da. Calle Poniente y 6ta. Av. Norte en el municipio de Chalchuapa, departamento de Santa Ana.

---

[2]Chalchuapa un lugar para visitar (2019, Octubre)
Recuperado de: http://culturachalchuapaneca.blogspot.com/2009/10/historia-de-chalchuapa.html

1.2.3 Organigrama Alcaldía Municipal de Chalchuapa

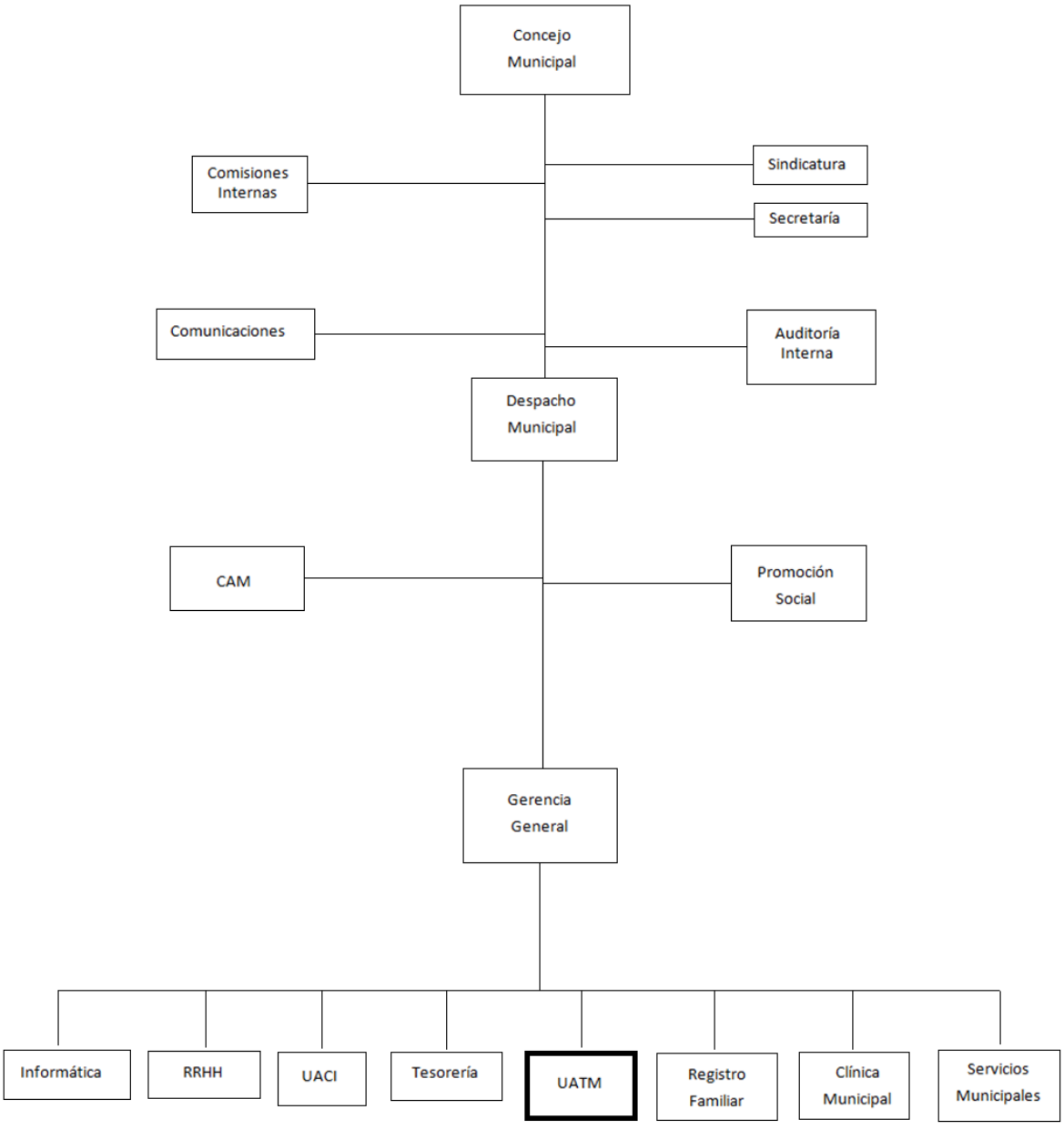


Figura 1: Organigrama Alcaldía Municipal de Chalchuapa.

1.2.4 Misión y Visión

Misión

Administrar los recursos provenientes del Estado, los generados por sus propios impuestos municipales y los provenientes de la ayuda por gestión internacional, de manera transparente y eficiente; para brindar servicios de alta calidad, que garanticen el bienestar de la comunidad Chalchuapaneca, a fin de garantizar que se cumplan los principios y objetivos del bien común con esmero, eficiencia y economía, de acuerdo a las áreas de su competencia.

Visión

Ser reconocidos como una institución de carácter público, eficiente, participativa e innovadora, que brinda alta calidad de servicios a los habitantes del municipio de Chalchuapa, mediante el óptimo aprovechamiento de sus recursos económicos y humanos motivados para el logro del bien común; desarrollando servicios y gestiones de calidad, aplicando principios institucionales de la nueva gerencia, que garanticen seguridad jurídica y bienestar social de la comunidad chalchuapaneca.

1.2.5 Servicios Municipales

Algunos de los servicios que presta la Alcaldía Municipal a la población chalchuapaneca son los siguientes: Cementerio Municipal, Biblioteca Municipal, Mercado Municipal, recolección de desechos sólidos (tren de aseo), Transporte Municipal, alumbrado público, Rastro Municipal, y Servicio de agua potable.

1.3 UNIDAD DE ADMINISTRACIÓN TRIBUTARIA MUNICIPAL

1.3.1 Descripción general

Su finalidad básica es atender al contribuyente o usuario en todo lo relacionado con los trámites de registro y control tributario, así como también verificar y controlar la precisión de la información declarada por los contribuyentes y usuarios, respetando para ello los derechos de estos y ejerciendo un asesoramiento pro-activo.

1.3.2 Funciones de la UATM

Algunas de las funciones de la Unidad de Administración Tributaria de la Alcaldía Municipal de Chalchuapa son:

- Dirigir y orientar la atención de las consultas que sean formuladas por usuarios o contribuyentes en materia de gestión de cobro.

- Se lleva el control de los registros de Cuentas Corrientes y sus saldos respectivos, que estos se lleven de forma confiable, completa y actualizada.

- Emisión de estados de cuentas y/o solvencias solicitados por contribuyentes y usuarios.

- Informar a contribuyentes y usuarios sobre la situación moratoria y gestionar el pago.

- Recolectar, registrar y actualizar la información de los propietarios, inmuebles y empresas para la determinación de los tributos municipales establecidos en la legislación correspondiente.

- Verificar y calificar los servicios que la municipalidad le presta a los inmuebles para su debido cobro.

- Notificar e instruir a los contribuyentes.

- Aplicar Ordenanzas y Leyes Tributarias con sus respectivas sanciones.

- Realizar monitoreos e inspecciones periódicas en las zonas asignadas para mantener el inventario de inmuebles y establecimientos actualizado.

- Llevar un registro y control de los distintos establecimientos dentro del municipio, clasificándolos por su tipología, teniendo en cuenta su ubicación, giro, representante legal, y solvencia de los mismos.

1.3.3 Servicio de agua potable en la UATM

El servicio de agua potable es de vital importancia para cualquier persona en el mundo, la municipalidad de Chalchuapa brinda este servicio a algunos sectores desde hace muchos años, cada vez con una mayor demanda.

Para tener derecho a este servicio las personas de la municipalidad deben hacer sus pagos en la Unidad de Administración Tributaria Municipal (UATM), en donde para hacer los cobros respectivos a dicho servicio se toma en cuenta si el contribuyente tiene un contador, el cual calcula los metros cúbicos consumidos, y así poder determinar el monto a cobrar, o si se tiene una cuota fija sin contador.

Para hacer sus pagos por el servicio de agua potable, el contribuyente siempre ha llegado a la Unidad de Administración Tributaria Municipal que cuenta con 3 áreas:

- Catastro
- Cuentas Corrientes
- Recuperación de Mora

El contribuyente siempre debe portar una boleta de cobro que es extendida por el área de Catastro para que este haga el pago correspondiente, la boleta es un papel que identifica al contribuyente por nombre para asignar la cuota respectiva, pero no hay sistema alguno que se integre con las demás áreas en este servicio.

El contribuyente presenta dicha boleta en el área de Cuentas corrientes (si no tiene mora) o en el área de recuperación de mora (si tiene mora), se busca a dicha persona en los archivos de la unidad y luego se envía al contribuyente al área de Tesorería, que recibe los pagos y genera los recibos de pago; esta área cuenta con SAFIMU (Sistema de Administración Financiero Integrado Municipal). Luego de haber pagado, el contribuyente regresa al área de Cuentas Corrientes para actualizar el pago, plantillas que se almacenan en archiveros, como un tipo de bitácoras y además un libro de Excel que maneja a los contribuyentes de cada sector que se beneficia del servicio de agua potable.

No existe una base de datos centralizada que contenga datos actualizados de los contribuyentes, o una forma en que Tesorería o SAFIMU actualicen los pagos al área de Cuentas Corrientes o Recuperación de Mora.

1.4 PLANTEAMIENTO DEL PROBLEMA

Uno de los objetivos principales de la Unidad de Administración Tributaria Municipal (UATM) de la Alcaldía de Chalchuapa es la de proporcionar a la población un servicio de calidad, rápido y eficaz.

Actualmente la problemática de esta unidad se debe en gran parte a que la información es procesada y registrada manualmente en documentos físicos, lo que no solamente causa una ineficiencia en la organización y clasificación de la información que procesa, sino que también retrasa el desarrollo de las distintas actividades de la unidad, así como también convierte la búsqueda de información y la generación de reportes y boletas en una tarea tediosa para el personal que maneja los cobros de agua.

Los problemas antes mencionados han dado lugar a una serie de efectos negativos a la administración. Uno de los principales es el tiempo de atención por parte de la unidad frente a las peticiones hechas por la población, esto se ve reflejado como retrasos al realizar búsquedas de personas que realizan pagos, o como retrasos en la rectificación y aplicación de pagos a los contribuyentes, ocasionando que la población tenga que esperar bastante tiempo para realizar sus diligencias.

Además, uno de los problemas que genera la falta de un sistema automatizado que lleve el control de los pagos, es que no existe modo de consultar de forma eficaz si una persona ya ha pagado o si todavía está pendiente de cancelar, lo que ocasiona que la alcaldía deje de recibir fondos, ya que permite que algunos usuarios del servicio puedan tener varios periodos de mora sin que el personal administrativo se percate de ello.

El deterioro de los registros físicos es otro problema, ya que muchas veces ocasiona pérdida total o parcial de la información, además de generar duplicidad de registros al querer reemplazar los archivos extraviados ya que no se cuenta con ningún otro respaldo de estos, lo que es un peligro latente en la Unidad de Administración Tributaria Municipal.

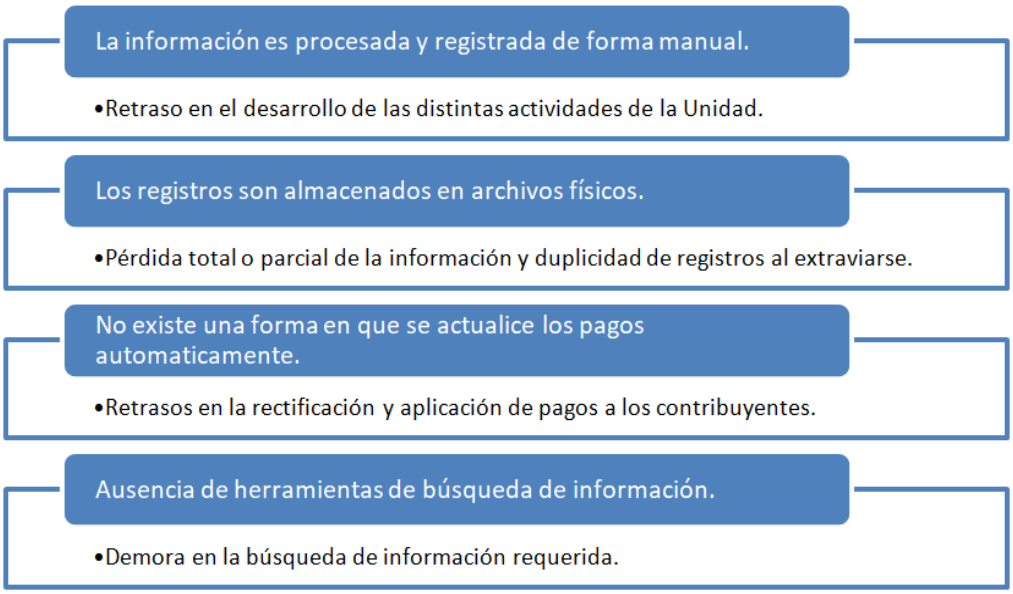


Figura 2: Diagrama Causa-Efecto UATM.

1.5 JUSTIFICACIÓN

Los procesos realizados manualmente, el estado de los registros físicos y la pérdida potencial de información importante son factores que producen a diario un retraso en las actividades diarias de la Unidad de Administración Tributaria de la Alcaldía Municipal de Chalchuapa en el tema de cobros por servicios de agua potable, por lo que se vuelve sumamente necesario contar con mecanismos que faciliten las labores del personal.

De esto resultan afectados no solo la Alcaldía Municipal de Chalchuapa, sino también los habitantes del municipio y sus alrededores que se benefician de este servicio. Por esto, la Unidad de Administración Tributaria Municipal (UATM) está en busca de un proceso de modernización que ayude a agilizar los procesos que ahí se realizan, por lo que se necesita disponer de una herramienta que automatice y facilite el registro y la obtención de información, así como también la generación automática de boletas de cobro para la pronta notificación a los contribuyentes del servicio de agua potable.

Se espera que las aglomeraciones de personas solo por preguntar si su pago ha sido realizado o para registrar su pago hecho hace solo unos minutos atrás se reduzca considerablemente, dejando mayor posibilidad al personal de realizar otras actividades en la unidad.

El sistema informático dará una solución de manera tecnológica a los problemas que la unidad administrativa sufre, logrando:

- Agilizar la obtención de información.

- Mejorar el control interno de los contribuyentes del servicio de agua potable de la UATM.

- Facilitar el manejo y clasificación de la información.

- Garantizar la seguridad y persistencia de los datos.

- Facilitar y automatizar la generación de boletas de cobro.

Además de ayudar a proyectar una imagen de calidad y eficiencia en los servicios que la Alcaldía Municipal de Chalchuapa brinda a las personas del municipio.

1.6 ALCANCES

El software para la administración y gestión de cobranzas del servicio de agua potable de la Alcaldía Municipal de Chalchuapa tendrá los siguientes alcances:

- El proyecto a realizar se centrará únicamente en los procesos relacionados al servicio municipal de agua potable.
- El sistema informático será independiente y no se enlazará con ningún otro sistema ya existente.
- La generación e impresión de boletas de cobro estará limitada a los procesos del servicio de agua potable.
- La implementación del sistema será de forma parcial, quedando solamente a nivel de pruebas del software.
- Se creará la respectiva documentación de operación de sistema, así como manuales de usuario y administrador.

1.7 LIMITACIONES

- La baja disponibilidad de tiempo del personal administrativo de la UATM para llevar a cabo una óptima recopilación de información y resolución de dudas que se puedan presentar durante el desarrollo del sistema, debido a sus deberes y responsabilidades de trabajo.

CAPÍTULO II

FACTIBILIDADES Y ANÁLISIS DE REQUERIMIENTOS

CAPÍTULO II FACTIBILIDADES Y ANÁLISIS DE REQUERIMIENTOS

2.1 METODOLOGÍA DEL DISEÑO

2.1.1 Ciclo de Vida del Software

Es una secuencia estructurada y bien definida de las etapas en Ingeniería de Software para desarrollar el producto deseado. Aporta una serie de pasos a seguir con la finalidad de diseñar y desarrollar un producto software de manera eficiente.

Hay varios modelos a seguir para el establecimiento de un proceso para el desarrollo de software, cada uno de los cuales describe un enfoque diferente para diferentes actividades que tienen lugar durante el proceso. Algunos autores consideran un modelo de ciclo de vida un término más general que un determinado proceso para el desarrollo de software.

El ciclo de vida más común y uno de los primeros modelos en surgir consta de siete fases: Análisis, Diseño, Desarrollo, Pruebas, Integración, Implementación y Mantenimiento.

Fase I: Análisis de requerimientos

En esta fase se analizan las necesidades de los usuarios finales del software para determinar qué objetivos debe cubrir. El equipo traza su plan e intenta crear el mejor y más conveniente modelo de software para el proyecto. De esta fase surge un documento llamado SRS (Software de Especificación de Requerimientos), que contiene la especificación completa de lo que debe hacer el sistema. En este documento, se establece una lista de los requerimientos acordados.

Fase II: Diseño del sistema

Descompone y organiza el sistema en elementos que puedan elaborarse por separado con la ayuda de toda la información recogida sobre requisitos y análisis. Las aportaciones de los usuarios y los resultados de la recogida de información hecha en la fase anterior serán las aportaciones base de la fase actual.

El resultado de esta etapa toma la forma de 2 diseños; El diseño lógico y el diseño físico. Los ingenieros crean meta-datos, diseñan la base de datos, crean diagramas de flujo de datos, las clases y en algunos casos pseudocódigos.

Fase III: Desarrollo del sistema

Esta fase también se puede denominar 'fase de programación'. La implementación del diseño de software empieza con el lenguaje de programación más conveniente, y desarrollando programas ejecutables y sin errores de manera eficiente.

Dependiendo del lenguaje de programación y su versión, se crean las bibliotecas y componentes reutilizables dentro del mismo proyecto para hacer que la programación sea un proceso mucho más rápido.

Fase IV: Pruebas del software

Los errores pueden arruinar el software tanto a nivel crítico y hasta el punto de ser eliminado. Esto incluye evaluación de módulos, evaluación del programa, evaluación del producto, evaluación interna y finalmente evaluación con el consumidor final.

Una parte de las pruebas de software la realizan solo los programadores, y otra la llevan a cabo de manera conjunta con los analistas de sistemas. Primero se realizan las pruebas con datos de muestra para determinar con precisión cuáles son los problemas y posteriormente se realiza otra con datos reales. Encontrar errores y su remedio a tiempo es la llave para conseguir un software fiable.

Fase V: Integración

El Software puede necesitar estar integrado con las bibliotecas, Bases de datos o con otro u otros programas. Esta fase del ciclo de vida se focaliza en la integración del software con las entidades del mundo exterior, se instala la aplicación en el sistema y se comprueba que funcione correctamente en el entorno en que se va a utilizar.

Fase VI: Implementación del sistema

Aquí se instala el software en máquinas de clientes. A veces, el software necesita instalar configuraciones para el consumidor final con posterioridad. En esta fase se capacita a los usuarios en el manejo del sistema, la capacitación la imparten los fabricantes, pero la supervisión de ésta es responsabilidad del analista de sistemas.

Fase VII: Mantenimiento del sistema

Esta fase confirma el funcionamiento del software en términos de más eficiencia y menos errores. Si se requiere, los usuarios se forman, o se les presta documentación sobre como operar y como mantenerlo en funcionamiento. Esta fase puede que tenga que encarar retos originados por virus ocultos o problemas no identificados del mundo real.

2.1.2 Modelo de Cascada

El modelo en cascada fue el primer modelo en ser introducido. Es también llamado modelo de ciclo de vida lineal-secuencial.

Es muy simple de entender y usar, ordena las etapas del proceso en una especie de pasos en forma descendente, de tal forma que el inicio de cada una de las etapas es subsecuente a la finalización de la anterior, por lo tanto no se puede avanzar a la siguiente etapa en el modelo sin que se haya completado satisfactoriamente la etapa anterior.

A continuación, las diferentes etapas del modelo:

Figura 3: Ciclo de Vida del Sistema: Modelo de Cascada.

Requisitos del Sistema. Identificación de problemas, determinación de los requerimientos y análisis de las necesidades del sistema.

Análisis del Sistema. El proceso de diseño traduce los requisitos en una representación del software con la calidad requerida antes de que comience la codificación.

Códigos. Desarrollo y documentación del software.

Pruebas. Pruebas del sistema para asegurar que la entrada definida produce los resultados que realmente se requieren.

Implementación. Implementación del sistema e instalación del software en el entorno que estará trabajando.

Operaciones y Mantenimiento. Mantenimiento y evaluación del sistema.

2.1.3 Ventajas y Desventajas del Modelo de Cascada

Ventajas

- Este modelo es muy simple y fácil de entender.
- Ayuda a la comprensión de los requisitos del proyecto.
- Refuerza los buenos hábitos: definir antes de diseñar, diseñar antes de codificar.
- Funciona bien con proyectos pequeños y proyectos donde los requerimientos están bien claros.

Desventajas

- No se puede tener el software funcionando hasta bien avanzado el ciclo de vida del proyecto.
- Cualquier error en el diseño que sea detectado en la etapa de pruebas puede conducir a un rediseño del sistema.

- Este modelo asume que todos los requisitos del cliente han sido completa y correctamente definidos al principio del proyecto, lo que hace muy difícil acomodarse a cualquier cambio en los requerimientos.

2.2 HERRAMIENTAS Y TÉCNICAS DE INVESTIGACIÓN

Una metodología es un conjunto integrado de técnicas y métodos cuya función básica radica en ser herramientas útiles al momento de recolectar datos e información importante, que luego de ser analizados detenidamente se convierten en información útil, veraz y oportuna que sirva para describir el problema y sus causas y a la vez producir soluciones efectivas.

Estas son algunas de las técnicas de investigación utilizadas para la recolección de datos e información para la formulación y posterior análisis de los requerimientos que facilitaran la ejecución de las distintas fases del ciclo de vida de un sistema:

2.2.1 Método de Observación Científica

La observación es una técnica que consiste en observar atentamente la situación y registrarla para su posterior análisis. Permite conocer la realidad mediante la percepción directa del objeto de investigación.

Puede utilizarse en distintas etapas de la investigación, es su etapa inicial se usa para diagnosticar el problema a resolver y para el posterior diseño de los requerimientos, y durante su desarrollo puede llegar a utilizarse para la comprobación de hipótesis. Al final de la investigación, la observación puede llegar a predecir tendencias y desarrollo de los fenómenos.

Además, la observación también permite al analista responder a las siguientes preguntas: que se está haciendo, quien lo está haciendo, como se está haciendo, cuando se hace, cuánto tiempo toma hacerlo, dónde se hace y por qué se hace.

Esta técnica se llevó a cabo durante todo el transcurso del proyecto en las unidades involucradas para observar con mayor claridad el funcionamiento de todos los procesos.

2.2.2 Entrevista

La entrevista es una técnica de recopilación de información mediante una conversación verbal dirigida con un propósito en específico. Quienes responden pueden ser gerentes o empleados, los cuales son usuarios actuales del sistema existente, usuarios potenciales del sistema propuesto o aquellos que proporcionarán datos o serán afectados por la aplicación propuesta.

Son los primeros contactos de la investigación, que pueden tratar de una serie de visitas entre el equipo desarrollador y los encargados que desean la solución informática para llevar a cabo la recopilación de información. La entrevista puede ser aplicada a todo tipo de persona, independientemente de la edad, nivel de educación o limitación física, ya que no necesita de una respuesta escrita.

Según el fin que se quiere conseguir con la entrevista, esta puede estar estructurada o no estructurada

Estructurada o cerrada

La entrevista estructurada es un tipo de entrevista personal, en la que el entrevistador utiliza un formato fijo, en el que las preguntas se preparan con anticipación.

Es un método de investigación cuantitativa utilizado para el propósito de la encuesta, cuyo objetivo es presentar las preguntas preestablecidas, en cada entrevista, en la misma secuencia.

No estructurada o abierta

Es un método de investigación cualitativo que no utiliza ningún formato fijo, sin embargo, el entrevistador puede tener algunas preguntas planificadas preparadas de antemano. En este formato, las preguntas se realizan de acuerdo a las respuestas que vayan surgiendo durante la entrevista. Como la entrevista no está planificada, tiene un enfoque informal donde se lleva a cabo una conversación amistosa entre el entrevistador y el entrevistado.

La entrevista de tipo no estructurada fue el tipo de técnica que se puso en práctica con los encargados de la UATM y demás personal administrativo, que son los que se beneficiaran directamente con el proyecto.

2.2.3 Investigación Documental

Este tipo de investigación, se base en consultar fuentes documentales o bibliográficas ya sean impresas o digitales que aporte información útil para la investigación.

Se puso en práctica este método consultando la "Ordenanza Reguladora de Tasas por Servicios Municipales" de la Alcaldía Municipal de Chalchuapa que es la que rige el servicio de agua potable. También se realizaron investigaciones por medio de sitios web y libros de texto, que ayudaron a solventar dudas durante el desarrollo del proyecto.

2.3 PLANTEAMIENTO DE LA SOLUCIÓN

Una vez analizadas las deficiencias que se presentan dentro de la Unidad de Administración Tributaria Municipal de la Alcaldía de Chalchuapa con respecto a la gestión del servicio de agua potable, se diseñará y desarrollará un sistema informático que tendrá los siguientes elementos ayudando a solventar las debilidades de la unidad:

- Registro y gestión de contribuyentes.
- Administración de proyectos de agua potable.
- Registro de los distintos servicios por agua potable.
- Ingreso de lecturas por servicio de medidor.
- Calculo de montos a los contribuyentes.
- Generación de boletas de cobro.
- Registro de pagos por cada servicio.
- Consulta de historial de pagos de cada contribuyente.

2.4 ESTUDIO DE FACTIBILIDAD

Después de definir la problemática y establecer las causas que hacen necesario un nuevo sistema, es pertinente realizar un estudio de factibilidad para determinar la viabilidad del desarrollo y la implementación del sistema en cuestión, así como los costos, beneficios y el grado de aceptación que la propuesta genera en la institución.

Un estudio de factibilidad bien ejecutado será clave para el éxito de un proyecto, puesto que el grado de factibilidad que se presente en cada aspecto determinará si el proyecto podrá o no realizarse desde los puntos de vista:

- Técnico
- Operativo
- Económico
- Legal.

2.4.1 Factibilidad Técnica

Este estudio está orientado a recolectar información sobre los componentes técnicos que posee la UATM de la municipalidad de Chalchuapa en cuanto a hardware y software mínimo para ejecutar el sistema. También indica si el equipo de desarrollo cuenta con el hardware y software necesario para desarrollar la aplicación.

Hardware y Software

Para el buen funcionamiento del sistema, es necesario que el software este manejado por hardware adecuado a las necesidades.

La Alcaldía de Chalchuapa cuenta con los siguientes recursos de hardware y software para la puesta en marcha del sistema:

Características del servidor	
Procesador	Intel Xeon 2.27 GHz 16 núcleos
Memoria RAM	20 GB DDR4
Almacenamiento	RAID 5 270 GB
Sistema Operativo	Windows Server 2012 R2
Características de los equipos de la UATM	
Procesador	Intel Core i5 3.3 GHz
Memoria RAM	4 GB
Unidad de Disco Duro	250 GB
Resolución de Pantalla	1366x768
Tarjeta de Red	Si
Impresora	Ricoh Aficio MP 2851
Sistema Operativo	Windows 7 x64
Navegador	Google Chrome
Visor de PDF	Adobe Reader

Tabla 1: Características de equipo hardware y software UATM.

Para el buen desarrollo de la aplicación, el equipo de desarrollo cuenta con las siguientes especificaciones tanto de software como de hardware:

Características del hardware	PC-01	PC-02	PC-03
Procesador	Intel Core i3 1.9 GHz	Intel Core i3 3.5 GHz	Intel Pentium Quad Core 2.4 GHz
Memoria RAM	6.0 GB	8.0 GB	4.0 GB
Unidad de Disco Duro	768 GB	1 TB	500 GB
Resolución de Pantalla	1366x768	1360x768	1366x768
Tarjeta de Red	Si	Si	Si
Wi-Fi	Si	No	Si

Tabla 2: Características hardware equipo de desarrollo.

Características del software	PC-01	PC-02	PC-03
Sistema Operativo	Windows 10 x64	Windows 10 x64	Windows 8.1 x64
Navegador	Google Chrome	Google Chrome	Google Chrome
Visor de PDF	Sumatra PDF	Adobe Reader	Adobe Reader
Gestor de Base de Datos	MySQL 5.7.24	MySQL 5.7.24	MySQL 5.7.24
Servidor de Aplicaciones	Apache 2.4.35	Apache 2.4.35	Apache 2.4.35

Tabla 3: Características software equipo de desarrollo.

Al evaluar el hardware y software existente tanto en la Alcaldía Municipal de Chalchuapa como el del equipo de desarrollo, se puede apreciar que no se requiere actualizar el equipo existente para la puesta en marcha del sistema ni para el desarrollo del mismo, ya que los equipos con lo que cuenta la alcaldía satisfacen los requerimientos establecidos para poner en funcionamiento del sistema propuesto.

2.4.2 Factibilidad Operativa

La factibilidad operativa de un proyecto informático radica en la capacidad de hacer uso del mismo en la fase de implementación y en el recurso humano involucrado.

Debe existir apoyo suficiente para el proyecto por parte de la administración, y personal capacitado para llevar a cabo el proyecto; en algunos casos puede que se evalúe la necesidad de capacitar al personal para que cumplan los requerimientos tecnológicos para la puesta en marcha del sistema.

Las entrevistas realizadas a los encargados y jefes de la Unidad de Administración Tributaria Municipal muestran que existe total aceptación del proyecto, ya que han expresado que existe la necesidad de contar con un sistema informático que facilite los procesos que se realizan en la unidad; los empleados también han manifestado su aceptación por el sistema, ya

que han experimentado muchas dificultades con la manera en la que actualmente se realizan las tareas.

Para el desarrollo del sistema se cuenta con un equipo de desarrollo conformado por 3 integrantes, cada uno tomará el rol conveniente para cada etapa en que se esté desarrollando el proyecto.

Los roles que se deben asumir se detallan a continuación:

Rol	Características
Analista	Habilidades para la obtención y análisis de información.
	Capacidad de análisis.
	Orientación al cliente.
	Comunicación efectiva.
Diseñador	Habilidades para la interpretación.
	Orientación al cliente.
	Capacidad de establecer estándares de diseño para aplicaciones web.
	Enfoque creativo.
Programador	Habilidades para el análisis de información.
	Manejo de diferentes lenguajes de programación web.
	Habilidades lógicas.
	Manejo de paradigmas de programación.
Administrador de Bases de Datos	Capacidad para diseñar bases de datos relacionales.
	Conocimiento y experiencia con gestores de bases de datos.

Tabla 4: Roles del equipo de desarrollo.

Como conclusión, se puede asegurar que el personal de la UATM posee los conocimientos básicos en el manejo de computadores y está bastante familiarizado con el uso de sistemas de información, por lo que el sistema es operativamente factible.

2.4.3 Factibilidad Económica

El objetivo de este apartado es hacer un estudio para determinar si el proyecto a desarrollar será factible económicamente o no. Conocer si un proyecto es factible económicamente demuestra si este puede o no llevarse a cabo o si va a generar más ganancias que costos al implementarlo dentro de la institución.

Costos de operación

La evaluación económica inicia conociendo los costos de operación que se van a generar para el desarrollo del sistema. El estudio revela que la Alcaldía Municipal de Chalchuapa no incurrirá en gastos adicionales de mobiliario, instalaciones eléctricas o personal adicional para la puesta en marcha el sistema, pues actualmente se cuenta con todos los insumos necesarios.

Debido a la naturaleza del proyecto, no se requiere de una inversión inicial para la implementación de este.

Costos de desarrollo

Son aquellos gastos en los que se incurre una tan sola vez, y se aplican en la creación de un bien. Incluye el costo de los materiales, mano de obra y los gastos indirectos de fabricación.

Es necesario calcular la depreciación de los equipos en los que se desarrollará el sistema. Según el Art. 30 de la Ley de Impuestos sobre la Renta, todo equipo electrónico está sujeto al 50% de depreciación anual sobre su valor.

Cada equipo utilizado para el desarrollo tiene la siguiente depreciación:

Equipo	Costo	Deprecación Anual	Depreciación durante el desarrollo (6 meses)
PC-01	$300.00	$150.00	$75.00
PC-02	$350.00	$175.00	$87.50
PC-03	$250.00	$125.00	$62.50
TOTAL	$900.00	$450.00	$225.00

Tabla 5: Depreciación del hardware del equipo de desarrollo.

El costo del kWh es de $0.19 IVA incluido[3]. Una laptop gasta en promedio aproximadamente 45 W por hora, mientras que un equipo de escritorio completo gasta en promedio 270 W por hora sumándole un monitor LCD que gasta un aproximado de 20 W la hora[4].

Cada equipo se utilizó un promedio de 5 horas al día, con lo cual los costos de energía eléctrica durante el periodo de desarrollo son los siguientes:

Equipo	kWh consumidos al mes	Costo kWh	Consumo durante el desarrollo (6 meses)
PC-01	6.75 kW	$0.19	$7.70
PC-02	40.50 kW	$0.19	$46.17
PC-03	6.75 kW	$0.19	$7.70
TOTAL	54.00 kW	$0.19	$61.57

Tabla 6: Consumo de energía de equipo de desarrollo.

[3]AES El Salvador, Tarifas vigentes del servicio eléctrico (2019, Julio)
Recuperado de: http://www.aes-elsalvador.com/site/assets/files/1168/tarifas-clesa.jpg

[4]How Many Watts Does a Computer Use? (2019, Julio)
Recuperado de: https://www.techwalla.com/articles/how-many-watts-does-a-computer-use

Costo del software para el desarrollo del sistema:

Descripción	Costo
Licencias Sistema Operativo	$0.00
MySQL 5.7.24	$0.00
Apache 2.4.35	$0.00
MySQL Workbench 8.0.16	$0.00
PHPMyAdmin 4.9.0.1	$0.00
PHP 7.2.20	$0.00
TOTAL	$0.00

Tabla 7: Costo del software para desarrollo.

El salario de cada uno de los desarrolladores y analistas se toma de $300 mensuales, esto restando los descuentos por préstamo de servicios. Tomando en cuenta lo anterior, durante los 6 meses que se trabajó en el proyecto el gasto en salarios sería el siguiente:

Rol	Salario mensual	Costo por salario durante el desarrollo (6 meses)
Analista	$350.00	$2,100.00
Diseñador	$350.00	$2,100.00
Programador	$350.00	$2,100.00
TOTAL		$5,670.00 (con descuentos)

Tabla 8: Salarios del equipo de desarrollo.

Costo total del desarrollo del proyecto:

Concepto	Costo
Depreciación de equipo de desarrollo	$225.00
Gastos de Electricidad	$61.57
Software de desarrollo	$0.00
Salario de los desarrolladores	$5,670.00
TOTAL	**$5,956.57**

Tabla 9: Costo total del desarrollo del proyecto.

De acuerdo al análisis de factibilidad económica se concluye que el precio total de la inversión del proyecto sería de $5,956.57.

Tomando en cuenta que la Alcaldía Municipal ya cuenta con el hardware necesario para la puesta en marcha del sistema y el costo por el desarrollo del sistema que se refleja en el presupuesto no correrá por parte de la alcaldía, puede decirse que la implementación del nuevo sistema es factible económicamente para la Alcaldía Municipal de Chalchuapa.

2.4.4 Factibilidad Legal

Todas las operaciones que se realicen dentro del sistema se deberán regir por la "Ordenanza Reguladora de Tasas por Servicios Municipales" de la Alcaldía Municipal de Chalchuapa. (Anexo 1: Ordenanza Reguladora de Tasas por Servicios Municipales).

Todas las computadoras de la UATM en las que se estará ejecutando el sistema poseen CD's de instalación con números de serie propios y licencias originales, estos no infringen ninguna ley nacional o internacional sobre derechos de autor. De igual manera, el software será desarrollado utilizando software libre y puede utilizarse sin incurrir en ilegalidad.

2.5 REQUERIMIENTOS DEL SISTEMA

Para un correcto funcionamiento del sistema a desarrollar, se precisa tomar en cuenta los requerimientos que reflejan las necesidades a satisfacer.

Los requerimientos para este proyecto se pueden clasificar en:

- Requerimientos Funcionales
- Requerimientos No Funcionales
- Requerimientos Operativos
- Requerimientos de Desarrollo
- Requerimientos Ambientales

2.5.1 Requerimientos Funcionales

Los requerimientos funcionales de un proyecto nos muestran la funcionalidad o servicios y tareas que se esperan que el sistema realice.

Autenticación y gestión de usuarios

- El sistema manejará niveles jerárquicos representados por su rol: administrador, cuentas corrientes, recuperación de mora y tesorería.
- El sistema deberá autenticar el acceso al sistema pidiendo un nombre de usuario y una contraseña que solo el usuario conoce.
- Los módulos a los que se tendrá acceso o mostrará el sistema deberán cambiar de acuerdo al usuario (rol).

- El sistema hará uso de contraseñas encriptadas para una mayor seguridad.
- El sistema manejará procesos de creación, modificación y eliminación de usuarios.
- Podrá mostrará vistas de todos los usuarios que manejan el sistema.

Administración de los módulos del sistema

- El sistema manejará procesos de creación, modificación y eliminación de proyectos de agua potable.
- El sistema manejará procesos de creación y eliminación de servicios de agua a cada contribuyente.
- El sistema podrá cambiar los estados de los servicios existentes (habilitar-deshabilitar).
- El sistema manejará procesos de creación, modificación y eliminación de contribuyentes (datos personales del cliente).

Actividades dentro del sistema

- El sistema podrá manejar los distintos tipos de servicio de agua que se prestan en la UATM: cuota fija, medidor y mecha. El sistema debe tener la capacidad de calcular los montos a pagar de acuerdo a cada uno de estos servicios.
- El sistema administrará fecha de inicio de facturación de cada contribuyente.
- El sistema mostrará vistas de todos proyectos creados.
- Podrá mostrar vistas de todos los contribuyentes existentes.

- Podrá mostrar vistas de todos los servicios por zona-proyecto.
- Podrá mostrar vistas de todos los pagos (historial de pagos) realizados o pendientes de los contribuyentes.
- El sistema deberá llevar el registro de las lecturas de medidor por cada servicio de medidor, así como también llevar el registro de todos los excedentes en las lecturas para poder hacer los cálculos de monto a pagar.
- El sistema administrará las fechas de vencimiento mensuales por el servicio de agua potable.
- El sistema tendrá una barra de búsqueda en todos los módulos para facilitar la búsqueda de contribuyentes específicos.
- El módulo de cuentas corrientes y recuperación de mora enviará a los contribuyentes a tesorería para que puedan realizar el pago.
- El sistema registrará los pagos realizados, guardando el código de recibo cancelado y la fecha y hora de cancelación.
- El sistema generara boletas de cobro con el monto a pagar de cada contribuyente por proyecto (zonas).

2.5.2 Requerimientos No Funcionales

Son todos aquellos requerimientos que nos son esenciales para el funcionamiento del sistema pero que ayudan o complementan a los funcionales.

Algunos de estos requerimientos son:

- Deberá tener una interfaz amigable y fácil de usar.
- El sistema tendrá que ser de ambiente web.
- El sistema deberá ser personalizado a la alcaldía de Chalchuapa: tener un nombre apegado a la unidad y los servicios que presta, y logos oficiales de la institución.
- Tendrá que mostrar el nombre de usuario activo en cada página de la aplicación.

2.5.3 Requerimientos Operativos

Los requerimientos operativos son todos aquellos que son fundamentales para que el sistema funcione correctamente y de forma óptima.

La alcaldía deberá contar con al menos los siguientes requisitos mínimos tanto en hardware como en software:

Características mínimas del servidor	
Procesador	Intel/AMD dos núcleos 2.0 GHz
Memoria RAM	2 GB +
Almacenamiento	50 GB +
Sistema Operativo	Windows Server 2012
Características mínimas de los equipos de la UATM	
Procesador	Intel/AMD uno o dos núcleos 2.0 GHz
Memoria RAM	2 GB +
Unidad de Disco Duro	50 GB +
Tarjeta de Red	Ethernet 100Mbps
Impresora	Inyección de tinta o láser
Sistema Operativo	Windows 7 o superior
Navegador	Google Chrome o Mozilla Firefox
Visor de PDF	Adobe Reader

Tabla 10: Requisitos mínimos hardware y software UATM.

2.5.4 Requerimientos de Desarrollo

Estos requerimientos comprenden aquellos recursos que deben de estar disponibles para desarrollar la aplicación o proyecto.

En la siguiente tabla se describen los requisitos mínimos de hardware con los que debe contar el equipo de trabajo para poder desarrollar el sistema:

Características mínimas del hardware	PC-01	PC-02	PC-03
Procesador	Intel/AMD 1.5 GHz	Intel/AMD 1.5 GHz	Intel/AMD 1.5 GHz
Memoria RAM	4.0 GB +	4.0 GB +	4.0 GB +
Unidad de Disco Duro	50 GB +	50 GB +	50 GB +
Conexión a internet	Ethernet 100Mbpsó Wi-Fi	Ethernet 100Mbpsó Wi-Fi	Ethernet 100Mbpsó Wi-Fi

Tabla 11: Requisitos mínimos hardware equipo de desarrollo.

En la siguiente tabla se listan los requisitos mínimos de software con los que debe contar el equipo de desarrollo:

Características mínimas del software	PC-01	PC-02	PC-03
Sistema Operativo	Windows 7 +	Windows 7 +	Windows 7 +
Navegador	Google Chrome o Mozilla Firefox	Google Chrome o Mozilla Firefox	Google Chrome o Mozilla Firefox
Visor de PDF	Adobe Reader u otros	Adobe Reader u otros	Adobe Reader u otros
Gestor de Base de Datos	MySQL	MySQL	MySQL
Servidor de Aplicaciones	Apache	Apache	Apache
Lenguaje de programación	PHP 5	PHP 5	PHP 5

Tabla 12: Requisitos mínimos software equipo de desarrollo.

2.5.5 Requerimientos de Seguridad Ambiental

Para que el sistema funcione correctamente hay que considerar factores ambientales tales como:

Ubicación.

- El equipo debe estar ubicado en un lugar donde no exista mucho movimiento de personal.
- Las computadoras deberán estar instaladas sobre escritorios o muebles estables, o especialmente diseñados para ello.
- El equipo deberá estar ubicado en lugares adecuados, lejos de la luz del sol y de ventanas abiertas.

Voltaje

- La energía eléctrica debe ser regulada a 110 voltios y con polo a tierra para evitar daño a los componentes internos del equipo.
- Deberá contar con reguladores de voltajes o UPS para evitar el daño a los equipos cuando haya un aumento del voltaje en las instalaciones.

Mantenimiento

- Se deberá mantener libre de polvo las partes externas de la computadora y de las impresoras para aumentar el rendimiento y la vida útil de los mismos.

CAPÍTULO III

DISEÑO DEL SISTEMA

CAPÍTULO III DISEÑO DEL SISTEMA

3.1 DISEÑO DE LA BASE DE DATOS

3.1.1 Bases de Datos

Una base de datos es una colección organizada de datos, generalmente almacenada, administrada y actualizada desde un sistema computacional. Los datos están organizados en filas, columnas y tablas, y esta indexada para hacer fácil el encontrar información relevante. Los datos se actualizan, expanden y borran como nueva información es añadida o modificada. Las bases de datos típicamente contienen datos tales como transacciones de ventas, catálogo de productos, inventarios y perfil de clientes.

Una base de datos está conformada de varios componentes principales:

Esquema. Una base de datos contiene uno o más esquemas, los cuales son básicamente una colección de una o más tablas de datos.

Tabla. Cada tabla contiene múltiples columnas, las cuales son similares a las columnas en una hoja de cálculo. Una tabla puede tener como mínimo dos columnas y tantas como cien o más, dependiendo del tipo de datos almacenados en ella.

Columna. Cada columna contiene uno de los varios tipos de datos o valores, como fecha (date), numéricos (integer, double) o valores alfanuméricos (varchar).

Fila. Los datos en una tabla están listados en filas, las cuales son como las filas en una hoja de cálculo. En la mayoría de los casos existen cientos o miles de filas de datos en una tabla.

Cada tabla en una base de datos relacional (a veces llamada entidad) contiene una o más categorías de datos o columnas también llamadas atributos. Cada fila, también llamada registro o tupla, contiene una única instancia de datos. Cada tabla contiene una única llave primaria, que identifica la información en una tabla.

Relaciones

En el modelo relacional, las tablas están relacionadas unas con otras. La relación entre tablas puede ser establecida por medio del uso de llaves foráneas (un campo en la tabla que la enlaza con la llave primaria de otra tabla). Las tres principales relaciones entre tablas son:

Relación uno-a-uno (1:1): Por ejemplo, cada cliente en una base de datos está asociado a una dirección de correo.

Relación uno-a-muchos (1:M): Por ejemplo, un cliente puede hacer una orden por múltiples productos. El cliente está asociado con múltiples entidades, pero todas esas entidades tienen una sola conexión de vuelta al mismo cliente.

Relación muchos-a-muchos (M:N): Por ejemplo, en una compañía donde todos los agentes trabajan con múltiples clientes, cada agente está asociado con múltiples clientes, y múltiples clientes pueden también estar asociados con múltiples agentes.

Un Gestor de Bases de Datos (SGBD) es el software que interactúa con el usuario final, aplicaciones y la base de datos misma para capturar y analizar datos. Sirve esencialmente como una interfaz o medio de comunicación entre la base de datos y el usuario final o los programas de aplicación, asegurándose que los datos sean consistentes y permanezcan de fácil acceso.

Los SGBD administran tres elementos importantes que son:

- Los datos.
- El motor de base de datos (que permite que los datos sean accedidos o modificados).
- El esquema, que define la estructura lógica de la base de datos.

SQL (Lenguaje de Consulta Estructurada) es un lenguaje de programación estandarizado que es usado para administrar bases de datos relacionales y realizar varias operaciones con los datos almacenados en ellos, tales operaciones pueden ser la inserción de datos, consultas, actualizaciones y borrado de datos; también sirve para la creación y modificación de esquemas y el control de acceso a los datos.

3.1.2 Diccionario de Datos

Tabla customers

Descripción: Esta tabla almacena los datos personales del contribuyente que solicita el servicio de agua potable.

Nombre del campo	Tipo de dato	PK	FK	NotNull	Descripción
id	BIGINT (20)	✓		✓	Identificador del contribuyente.
firstname	VARCHAR (60)			✓	Nombres del contribuyente.
lastname	VARCHAR (60)			✓	Apellidos del contribuyente.
DUI	CHAR (10)			✓	Numero de DUI del contribuyente.
NIT	CHAR (17)				Numero de NIT del contribuyente.
phone	CHAR (9)				Número de teléfono del contribuyente.

Tabla 13: Diccionario de Datos – Tabla customers.

Tabladefeat_of_date_per_zones

Descripción: Esta tabla almacena las fechas límite de pago para todos los contribuyentes de cada zona o proyecto.

Nombre del campo	Tipo de dato	PK	FK	NotNull	Descripción
id	BIGINT (20)	✓		✓	Identificador de la fecha limite por zona.
defeatOfDate	DATE			✓	Fecha límite para pagar el mes anterior.
active	TINYINT (3)			✓	Indica si el registro pertenece al mes actual.
zone	BIGINT(20)		✓	✓	Zona a la que se le asignara fecha límite de pago.

Clave foránea	Campos	Referencia Tabla
defeat_of_date_per_zones_zone_fk	zone	zones

Tabla 14: Diccionario de Datos – Tabla defeat_of_date_per_zones.

Tabla history_water_services

Descripción: Se guarda el historial de cada contribuyente cada vez que ha habido una actualización en su servicio, esto es: si ha habido una desconexión o reconexión en su servicio de agua.

Nombre del campo	Tipo de dato	PK	FK	NotNull	Descripción
id	BIGINT (20)	✓		✓	Identificador del cambio de estado del servicio.
observation	TEXT				Explicación del porqué de la desconexión-reconexión.
statusWaterService	BIGINT (20)		✓	✓	Llave foránea que indica el estado del servicio.
waterService	BIGINT (20)		✓	✓	Llave foránea que indica el servicio que cambiará de estado.

Clave foránea	Campos	Referencia Tabla
history_water_services_statuswaterservice_fk	statusWaterService	status_water_services
history_water_services_waterservice_fk	waterService	Water_services

Tabla 15: Diccionario de Datos – Tabla history_water_services.

Tabla month_collectors

Descripción: Se almacenan los valores de acumulado correspondientes a cada mes de atraso de cada contribuyente, estos valores son necesarios para hacer el cálculo de la mora.

Nombre del campo	Tipo de dato	PK	FK	NotNull	Descripción
month	BIGINT (20)	✓		✓	Numero de meses que puede estar en mora.
collector	SMALLINT (5)			✓	Valor de acumulado correspondiente a los meses.

Tabla 16: Diccionario de Datos – Tabla month_collectors.

Tabla payments

Descripción: En esta tabla se almacenan todos los pagos de cuota mensual realizados por el contribuyente.

Nombre del campo	Tipo de dato	PK	FK	NotNull	Descripción
id	BIGINT (20)	✓		✓	Identificador de los pagos del contribuyente.
receipt	BIGINT (20)			✓	Numero de recibo de pago.
defeatOfDate	DATE			✓	Indica la fecha límite para realizar el pago.
amount	DOUBLE			✓	Monto a cancelar.
previousPayment	DATE			✓	Indica la fecha anterior a la última fecha de pago.
recentPayment	DATE			✓	Indica la última fecha de pago.
month	DOUBLE			✓	Contador de meses para calcular monto.

recentExcess	SMALLINT (5)			✓	Campo que almacena excesos en lecturas de medidor.
waterService	BIGINT (20)		✓	✓	Llave foránea que almacena que indica el servicio.
paymentStatus	BIGINT (20)		✓	✓	Llave foránea que indica el estado actual del pago.

Clave foránea	Campos	Referencia Tabla
payments_paymentstatus_fk	paymentStatus	payment_statuses
payments_watersevices_fk	waterService	water_services

Tabla 17: Diccionario de Datos – Tabla payments.

Tabla payment_statuses

Descripción: Esta tabla almacena los distintos estados por los que pasa un pago, que son: iniciado, en proceso, pendiente y finalizado.

Nombre del campo	Tipo de dato	PK	FK	NotNull	Descripción
id	BIGINT (20)	✓		✓	Identificador de estado de pago.
name	VARCHAR (60)			✓	Nombre de estado de pago.

Tabla 18: Diccionario de Datos – Tabla paymet_statuses.

Tabla status_water_services

Descripción: En esta tabla se almacenan los distintos estados de cada uno de los servicios de agua, los cuales son: apertura, desconexión y reconexión.

Nombre del campo	Tipo de dato	PK	FK	NotNull	Descripción
id	BIGINT (20)	✓		✓	Identificador de estado de servicio.
name	VARCHAR (60)			✓	Nombre de estado de servicio.

Tabla 19: Diccionario de Datos – Tabla status_water_services.

Tabla type_of_opening

Descripción: En esta tabla se guardan todos los tipos de apertura al solicitar un servicio de agua, estos son dos: con rompimiento y sin rompimiento.

Nombre del campo	Tipo de dato	PK	FK	NotNull	Descripción
id	BIGINT (20)	✓		✓	Identificador de tipo de apertura.
name	VARCHAR (60)			✓	Nombre del tipo de apertura del servicio.

Tabla 20: Diccionario de Datos – Tabla type_of_opening.

Tabla type_of_water_services

Descripción: Esta tabla almacena los distintos tipos de servicio de agua potable que la Alcaldía Municipal presta a las personas.

Nombre del campo	Tipo de dato	PK	FK	NotNull	Descripción
id	BIGINT (20)	✓		✓	Identificador del tipo de servicio.
name	VARCHAR (60)			✓	Nombre del tipo de servicio.
code	VARCHAR (10)			✓	Código contable del tipo de servicio.
charge	DOUBLE			✓	Monto asignado a cada servicio.

Tabla 21: Diccionario de Datos – Tabla type_of_water_services.

Tabla type_users

Descripción: Esta tabla guarda los distintos tipos de usuario que manejarán el sistema.

Nombre del campo	Tipo de dato	PK	FK	NotNull	Descripción
id	BIGINT (20)	✓		✓	Identificador del tipo de usuario.
name	VARCHAR (60)			✓	Nombre del tipo de usuario.

Tabla 22: Diccionario de Datos – Tabla type_users.

Tabla users

Descripción: En esta tabla se almacena el nombre y contraseña de los usuarios con permiso a acceder al sistema.

Nombre del campo	Tipo de dato	PK	FK	NotNull	Descripción
id	BIGINT (20)	✓		✓	Identificador del usuario.
name	CHAR (10)			✓	Nombre del usuario.
password	VARCHR (60)			✓	Clave de acceso del usuario.
typeUser	BIGINT (20)		✓	✓	Indica el rol del usuario.

Clave foránea	Campos	Referencia Tabla
users_typeuser_fk	typeUser	type_users

Tabla 23: Diccionario de Datos – Tabla users.

Tabla water_services

Descripción: Tabla que almacena todas las contrataciones por servicio de agua de cada contribuyente.

Nombre del campo	Tipo de dato	PK	FK	NotNull	Descripción
id	BIGINT (20)	✓		✓	Identificador de servicio.
placeWaterService	VARCHAR (255)			✓	Dirección del servicio a solicitar.
recentPayment	DATE			✓	Indica la última fecha de pago.
recentExcess	SMALLINT (5)			✓	Lectura de medidor reciente.
code	VARCHAR (255)			✓	Código del medidor.
typeOfOpening	BIGINT (20)		✓	✓	Tipo de apertura.
typeOfWaterService	BIGINT (20)		✓	✓	Tipo de servicio de agua.
zone	BIGINT (20)		✓	✓	Zona a la que pertenece el servicio.
customer	BIGINT (20)		✓	✓	Identifica al contribuyente
statusWaterService	BIGINT (20)		✓	✓	Estado del servicio de agua.

Clave foránea	Campos	Referencia Tabla
water_services_customer_fk	customer	customers
water_services_statuswaterservice_fk	statusWaterService	status_water_service
water_services_typeofopening_fk	typeOfOpening	type_of_opening
water_services_typeofwaterservice_fk	typeOfWaterService	type_of_water_service
water_services_zone_fk	zone	zones

Tabla 24: Diccionario de Datos – Tabla water_services.

Tabla water_service_type_consumes

Descripción: Esta tabla almacena las lecturas mensuales de cada contador.

Nombre del campo	Tipo de dato	PK	FK	NotNull	Descripción
id	BIGINT (20)	✓		✓	Identificador del consumo por servicio.
previousReading	BIGINT (20)			✓	Lectura anterior del contador.
currentReading	BIGINT (20)			✓	Lectura actual del contador.
waterService	BIGINT (20)		✓	✓	Indica a que servicio asignar las lecturas.
active	TINYINT (3)			✓	Indica si la medición pertenece al mes actual.

Clave foránea	Campos	Referencia Tabla
water_service_type_consumes_waterservice_fk	waterService	water_services

Tabla 25: Diccionario de Datos – Tabla water_service_type_consumes.

Tabla zones

Descripción: Se guardan en esta tabla todas las zonas (o proyectos) que se benefician del servicio de agua potable.

Nombre del campo	Tipo de dato	PK	FK	NotNull	Descripción
id	BIGINT (20)	✓		✓	Identificador de la zona.
description	VARCHAR (255)			✓	Nombre de la zona.

Tabla 26: Diccionario de Datos – Tabla zones.

3.1.3 Diagrama Entidad Relación

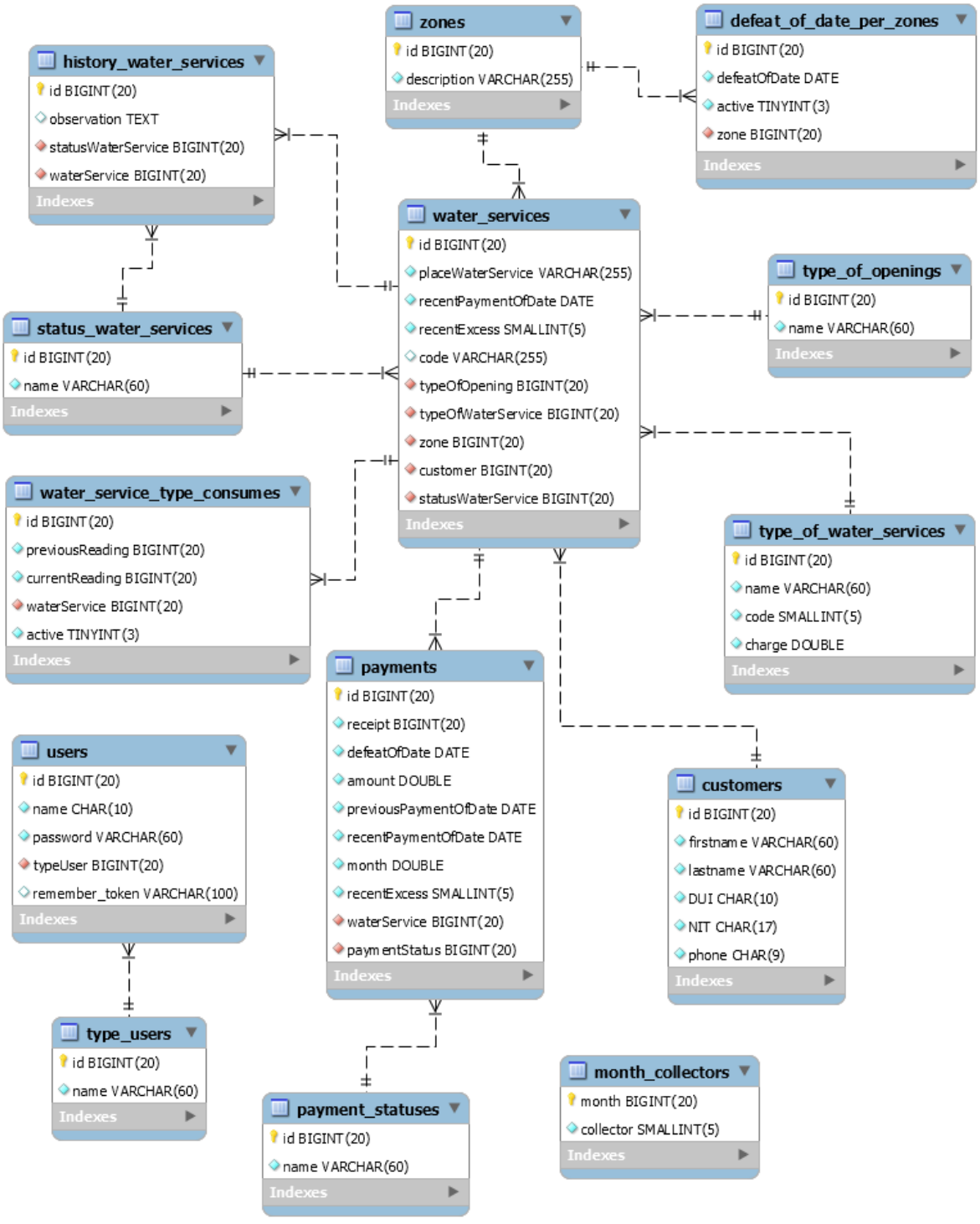


Figura 4: Diagrama ER de la Base de Datos.

3.2 DIAGRAMAS DE CASOS DE USO

3.2.1 Casos de Uso

Un diagrama de casos de uso, es un diagrama de comportamiento en UML (Lenguaje Unificado de Modelado). Los casos de uso modelan la funcionalidad de un sistema usando actores y casos de uso. Son una serie de acciones, servicios y funciones que el sistema necesita realizar. En este contexto, "sistema" se refiere a algo que está siendo desarrollado u operado, como un sitio web. Los "actores" son personas o entidades operando bajo roles definidos dentro del sistema.

Los diagramas de caso de uso son útiles para la visualización de los requerimientos funcionales de un sistema que se traducirá en elecciones de diseño y prioridades de desarrollo. También ayudan a identificar cualquier factor interno o externo que pueda influir en el sistema y que debería de tomarse en consideración.

Estos proveen de un alto nivel de análisis desde fuera del sistema. Los diagramas de casos de uso especifican como el sistema interactúa con los actores sin preocuparse sobre los detalles de cómo dicha funcionalidad es implementada.

Un diagrama de casos de uso efectivo puede ayudar al equipo a discutir y representar:

- Escenarios en los cuales el sistema o aplicación interactúa con personas, organizaciones o sistemas externos.
- Formas en las cuales el sistema o aplicación ayuda a las entidades (actores) a cumplir sus metas.
- El enfoque general del sistema.

La simbología básica de los diagramas de casos de uso es la siguiente:

Paquete

Se dibuja las fronteras del sistema usando un rectángulo que contiene el caso. Se colocan a los actores fuera de las fronteras del sistema. Un paquete también puede referirse como escenario.

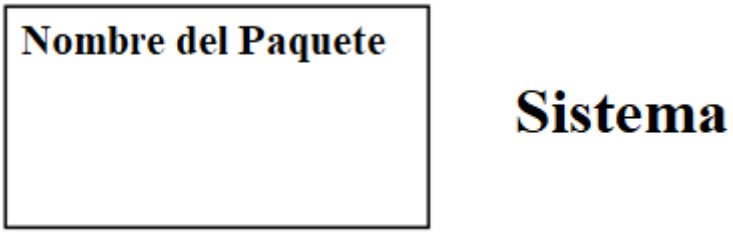


Figura 5: Casos de Uso – Paquete.

Caso de uso

Se dibujan los casos usando óvalos. Se etiquetan los óvalos con verbos que representan las funciones del sistema. En la mayoría de los casos, los casos de uso representan el objetivo final del actor.

Figura 6: Casos de Uso – Caso de Uso.

Actores

Los actores son los usuarios que interactúan con el sistema. Un actor puede ser una persona, una organización u otro sistema externo. Cuando un sistema es el actor de otro sistema, se etiqueta el sistema con el estereotipo de actor.

Figura 7: Casos de Uso – Actor.

Relaciones

Se ilustran las relaciones entre actor y el caso de uso con una simple línea. Para las relaciones entre casos de uso, se utilizan flechas etiquetadas ya sea con la palabra "include" o con la palabra "extend".

Las relaciones "include" indican que ese caso de uso es necesitado por otro para poder realizar una acción. Una relación "extend" indica opciones alternas bajo cierto caso de uso.

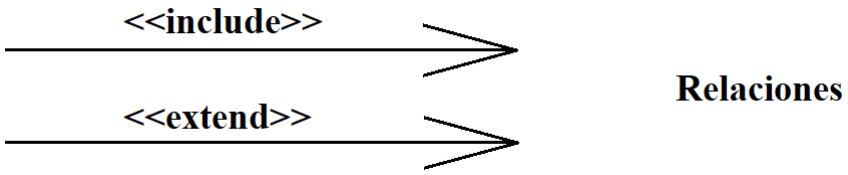


Figura 8: Casos de Uso – Relaciones.

3.2.2 Diagrama de Herencia de Usuario

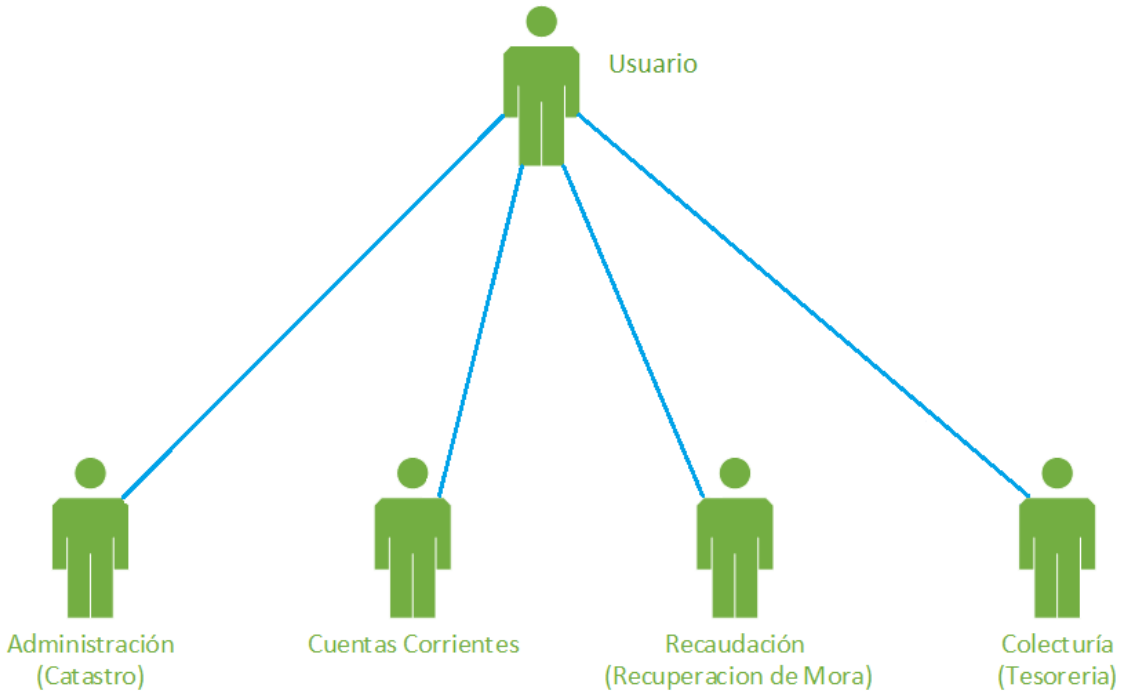


Figura 9: Diagrama de Herencia de Usuario.

3.2.3 Diagramas de Casos de Uso

Cada caso de uso tiene una descripción que lo describe detalladamente, a la descripción de estos se les describe como: escenarios de caso de uso. El caso de uso principal representa el flujo de los eventos en el sistema y las rutas alternativas describen las variaciones para el comportamiento.

A continuación, se presentan los diagramas de casos de uso y sus respectivos escenarios de casos de uso, del sistema propuesto.

Caso de Uso Acceder al Sistema

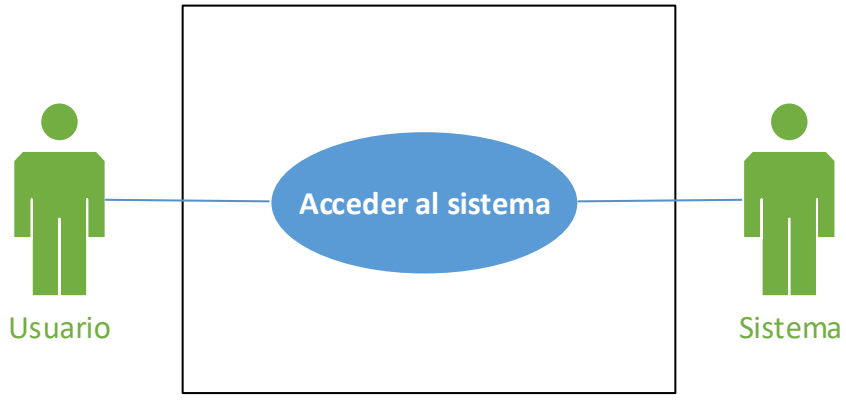


Figura 10: Caso de Uso – Acceder al Sistema.

Caso de uso	#1
Nombre de caso de uso:	ACCEDER AL SISTEMA
Actor Principal	Usuario
Actor secundario	Sistema
Descripción	Acceso al sistema por un usuario mediante la confirmación de sus credenciales en un formulario de autenticación y mostrar notificaciones de éxito o advertencia de ser necesario.
Activar evento	El usuario debe dar clic en el botón "Entrar" del formulario de login.
Secuencia Principal	1. El usuario accede al sistema mediante un navegador web. 2. Ingresa a la página de inicio de sesión. 3. Introduce las credenciales de "Usuario" y "Contraseña". 4. Si el usuario o contraseña son correctos ir al paso 5, de lo contario ir al paso 1. 5. Ir al caso según tipo de usuario.
Precondición	El sistema debe de estar activo.
Poscondición	Ingresar al sistema.

Tabla 27: Caso de Uso – Acceder al Sistema.

Caso de Uso Administrar Usuarios

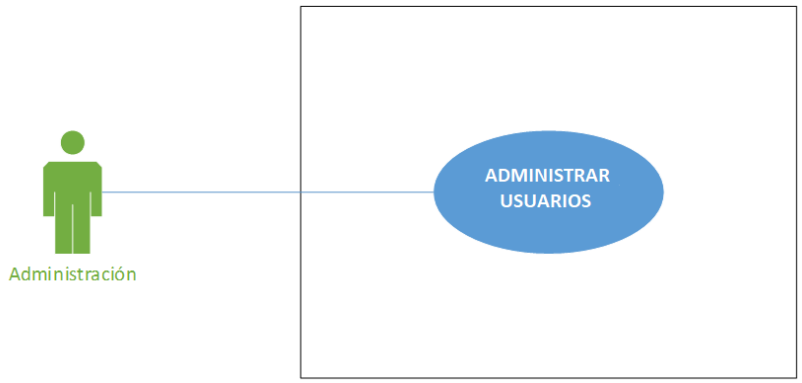


Figura 11: Caso de Uso – Administrar Usuarios.

Caso de uso	#2
Nombre de caso de uso:	ADMINISTRAR USUARIOS
Actor Principal	Administración
Descripción	Gestión de la información relacionada a los usuarios del sistema.
Activar evento	El usuario administrador debe dar clic en la sección “Usuarios” del panel de la vista de administración.
Secuencia Principal	1. Mostrar lista de usuarios si existen y un formulario para nuevo usuario. 2. Si “Nuevo Usuario”, saltar a paso 3, si “Editar” usuario ir al paso 5, “Eliminar” usuario ir al paso 9, “Salir” ir al paso 13. 3. Al crear un “Nuevo Usuario” digitar la información del nuevo usuario, “Usuario”, “Contraseña” y elegir el “Tipo” de usuario de la lista. 4. Dar clic en el botón “Guardar” y este se mostrará en la lista de usuarios inmediatamente. 5. Para “Editar” un usuario se debe identificar un usuario de la lista de acuerdo al criterio de búsqueda seleccionado. 6. Dar clic en el botón “Acción” el cual desplegara la opción de “Editar” y “Eliminar”, dar clic en el ítem “Editar”. 7. Se cargan el nombre de usuario en formulario editar listo

	para editar los campos de interés. 8. Dar clic en el botón "Guardar" y este se actualizará en la tabla de usuarios inmediatamente. 9. Para "Eliminar" un usuario se debe identificar un usuario de la lista de acuerdo al criterio de búsqueda seleccionado 10. Dar clic en el botón "Acción" el cual desplegara la opción de "Editar" y "Eliminar", dar clic en el ítem "Eliminar". 11. Mostrar mensaje de confirmación con las opciones "Cancelar" y "Confirmar". 12. Si "Confirmar", elimina el usuario, ir al paso 1; si "Cancelar" cancela la eliminación, ir al paso 1. 13. Para "Salir" ir al caso "Acceder al Sistema".
Precondición	Se debe autenticar como tipo de usuario "Administrador" al momento de ingresar al sistema.
Poscondición	Almacenar la hora y fecha de creación y actualización de usuario en la base de datos.

Tabla 28: Caso de Uso – Administrar Usuarios.

Caso de Uso Administrar Proyectos

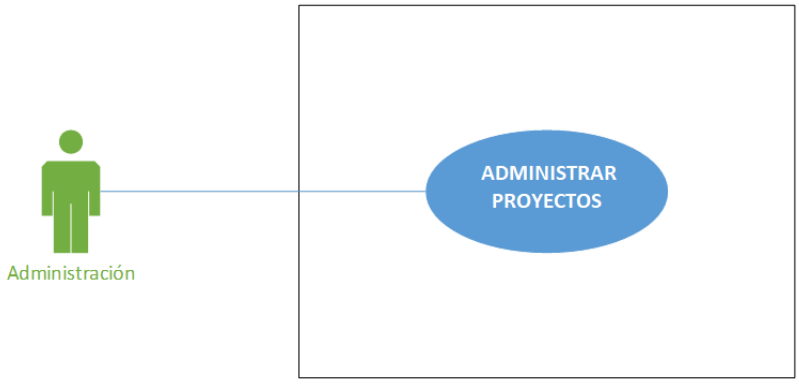


Figura 12: Caso de Uso – Administrar Proyectos.

Caso de uso	#3
Nombre de caso de uso:	ADMINISTRAR PROYECTOS
Actor Principal	Administración
Dependencia	Acceder al sistema
Descripción	Manejo de la información relacionada con los proyectos de distribución de agua potable llevados a cabo por la Alcaldía Municipal de Chalchuapa.
Activar evento	El usuario administrador debe dar clic en la sección "Proyectos" del panel de la vista de administración.
Secuencia Principal	1. Mostrar lista de proyectos si existen y un formulario para "Nuevo proyecto". 2. Si "Nuevo Proyecto", saltar a paso 3, si "Editar" ir al paso 5, "Eliminar" proyecto ir al paso 9, "Salir" ir al paso 13. 3. Al crear un "Nuevo proyecto" digitar la información del nuevo proyecto, "Nombre". 4. Dar clic en el botón "Guardar" y este se mostrará en la lista de proyectos inmediatamente. 5. Para "Editar" un proyecto se debe identificar un proyecto de la lista de acuerdo al criterio de búsqueda seleccionado. 6. Dar clic en el botón "Acción" el cual desplegara la opción de "Editar" y "Eliminar", dar clic en el ítem "Editar". 7. Se cargan el nombre del proyecto en formulario editar proyecto, listo para editar el campo. 8. Dar clic en el botón "Guardar" y este se actualizará en la tabla de proyectos inmediatamente. 9. Para "Eliminar" un proyecto se debe identificar un proyecto de la lista de acuerdo al criterio de búsqueda seleccionado. 10. Dar clic en el botón "Acción" el cual desplegara la opción de "Editar" y "Eliminar", dar clic en el ítem "Eliminar". 11. Mostrar mensaje de confirmación con las opciones "Cancelar" y "Confirmar". 12. Si "Confirmar", elimina el proyecto, ir al paso 1; si

	"Cancelar" cancela la eliminación, ir al paso 1. 13. Para "Salir" ir al caso "Acceder al Sistema".
Precondición	Se debe autenticar como tipo de usuario "Administrador" al momento de ingresar al sistema.
Poscondición	Almacenar la hora y fecha de creación y actualización de un proyecto en la base de datos.

Tabla 29: Caso de Uso – Administrar Proyectos.

Caso de Uso Administrar Tipo de Servicio

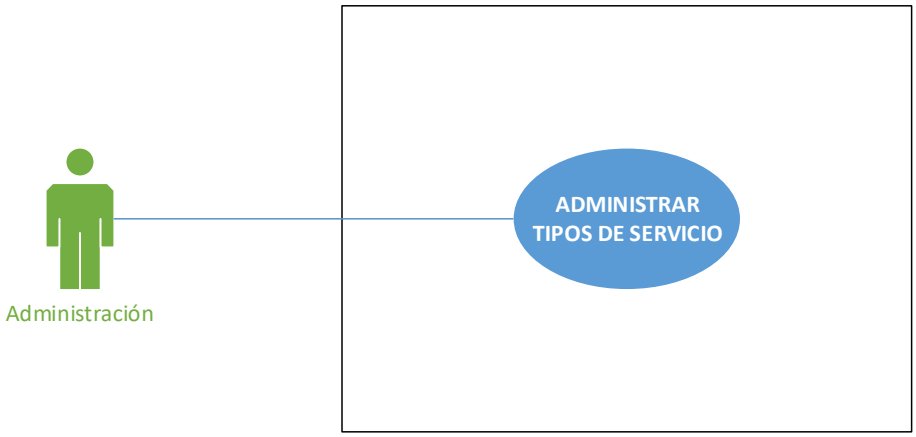


Figura 13: Caso de Uso – Administrar Tipos de Servicio.

Caso de uso	#4
Nombre de caso de uso:	ADMINISTRAR TIPOS DE SERVICIO
Actor Principal	Administración
Dependencia	Acceder al sistema
Descripción	Manejo de la información relacionada con los tipos de servicio de distribución de agua potable llevados a cabo por la Alcaldía Municipal de Chalchuapa.
Activar evento	El usuario administrador debe dar clic en la sección "Tipos de servicio" del panel de la vista de administración.
Secuencia Principal	1. Mostrar lista de tipos de servicio si existen y un formulario para "Nuevo tipo de servicio". 2. Si "Nuevo" tipo de servicio, saltar a paso 3, si "Editar" ir al

	paso 5, “Eliminar” tipo de servicio ir al paso 9, “Salir” ir al paso 13. 3. Al crear un “Nuevo” tipo de servicio, digitar la información del nuevo proyecto, “Tipo de servicio”, “Código contable” y “Monto”. 4. Dar clic en el botón “Guardar” y este se mostrará en la lista de tipos de servicio inmediatamente. 5. Para “Editar” un tipo de servicio se debe identificar un proyecto de la lista de acuerdo al criterio de búsqueda seleccionado. 6. Dar clic en el botón “Acción” el cual desplegara la opción de “Editar” y “Eliminar”, dar clic en el ítem “Editar”. 7. Se cargan el nombre del tipo de servicio en formulario editar proyecto, listo para editar los campos de interés. 8. Dar clic en el botón “Guardar” y este se actualizará en la tabla de tipos de servicio inmediatamente. 9. Para “Eliminar” un tipo de servicio se debe identificar un proyecto de la lista de acuerdo al criterio de búsqueda seleccionado 10. Dar clic en el botón “Acción” el cual desplegara la opción de “Editar” y “Eliminar”, dar clic en el ítem “Eliminar”. 11. Mostrar mensaje de confirmación con las opciones “Cancelar” y “Confirmar”. 12. Si “Confirmar Eliminar”, elimina el tipo de servicio, ir al paso 1; si “Cancelar” cancela la eliminación, ir al paso 1.
Precondición	Se debe autenticar como tipo de usuario “Administrador” al momento de ingresar al sistema.
Poscondición	Almacenar la hora y fecha de creación y actualización de un tipo de servicio en la base de datos.

Tabla 30: Caso de Uso – Administrar Tipos de Servicio.

Caso de Uso Administrar Contribuyentes

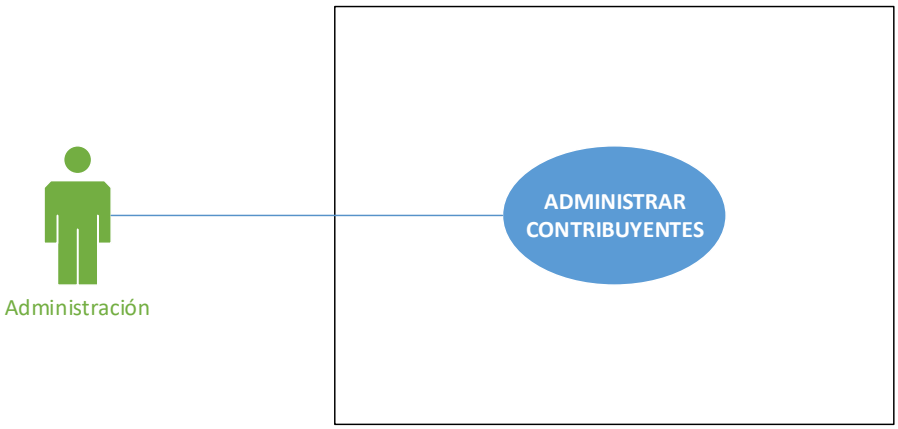


Figura 14: Caso de Uso – Administrar Contribuyentes.

Caso de uso	#5
Nombre de caso de uso:	ADMINISTRAR CONTRIBUYENTES
Actor Principal	Administración
Dependencia	Acceder al sistema
Descripción	Manejo de la información relacionada con los contribuyentes del servicio de distribución de agua potable llevado a cabo por la Alcaldía Municipal de Chalchuapa.
Activar evento	El usuario administrador debe dar clic en la sección "Contribuyentes" del panel de la vista de administración.
Secuencia Principal	1. Mostrar lista de contribuyentes si existen y un formulario para "Nuevo contribuyente". 2. Si "Nuevo" contribuyente, saltar a paso 3, si "Editar" ir al paso 5, "Eliminar" contribuyente ir al paso 9, "Salir" ir al paso 13. 3. Al crear un "Nuevo" contribuyente, digitar la información del nuevo contribuyente, "Nombres", "Apellidos", "Teléfono", "DUI" y "NIT". 4. Dar clic en el botón "Guardar" y este se mostrará en la lista de tipos de contribuyentes inmediatamente. 5. Para "Editar" un contribuyente se debe identificar un

	contribuyente de la lista de acuerdo al criterio de búsqueda seleccionado. 6. Dar clic en el botón “Acción” el cual desplegara la opción de “Editar”, “Eliminar” y “Nuevo servicio”, dar clic en el ítem “Editar”; si “Nuevo servicio”, ir al caso “Administrar servicios de agua”. 7. Se cargan los datos del contribuyente en formulario editar contribuyente, listo para editar los campos de interés. 8. Dar clic en el botón “Guardar” y este se actualizará en la tabla de contribuyentes inmediatamente. 9. Para “Eliminar” un contribuyente se debe identificar un contribuyente de la lista de acuerdo al criterio de búsqueda seleccionado 10. Dar clic en el botón “Acción” el cual desplegara la opción de “Editar”, “Eliminar” y “Nuevo servicio”, dar clic en el ítem “Eliminar”; si “Nuevo servicio”, ir al caso “Administrar servicios de agua”. 11. Mostrar mensaje de confirmación con las opciones “Cancelar” y “Confirmar”. 12. Si “Confirmar”, elimina el contribuyente, ir al paso 1; si “Cancelar” cancela la eliminación, ir al paso 1. 13. Para “Salir” ir al caso “Acceder al Sistema”.
Precondición	Se debe autenticar como tipo de usuario “Administrador” al momento de ingresar al sistema.
Poscondición	Almacenar la hora y fecha de creación y actualización de un contribuyente en la base de datos.

Tabla 31: Caso de Uso – Administrar Contribuyentes.

Caso de Uso Administrar Servicio de Agua

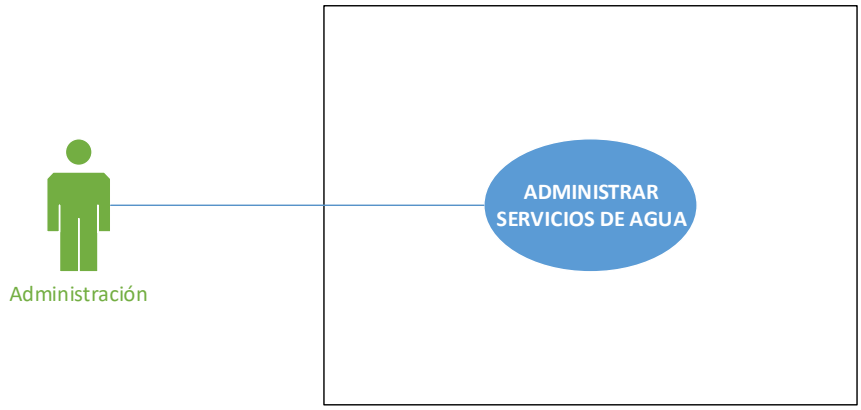


Figura 15: Caso de Uso – Administrar Servicios de Agua.

Caso de uso	#6
Nombre de caso de uso:	ADMINISTRAR SERVICIOS DE AGUA
Actor Principal	Administración
Dependencia	Acceder al sistema
Descripción	Manejo de la información relacionada con los servicios de agua proporcionados a los contribuyentes del servicio de distribución de agua potable.
Activar evento	El usuario administrador debe dar clic en la sección "Servicios de agua" del panel de la vista de administración.
Secuencia Principal	1. Mostrar lista de servicios de agua si existen con un botón de "Acción" para "Editar" y "Eliminar" un servicio específico. 2. Si "Nuevo Servicio" de agua, saltar a paso 3, si "Editar" ir al paso 7, "Eliminar" servicio de agua ir al paso 9, "Salir" ir al paso 13. 3. Para crear un "Nuevo servicio" de agua, ir a la sección "Contribuyentes" del panel de la vista de administración, identificar un contribuyente de la lista de acuerdo al criterio de búsqueda seleccionado. 4. Dar clic en el botón "Acción" el cual desplegara la opción de "Editar", "Eliminar" y "Nuevo servicio", dar clic en el

	ítem “Nuevo servicio”; si “Editar” o “Eliminar”, ir al caso “Administrar contribuyentes”. 5. Se carga en el formulario los campos e ítems “Tipo de apertura”, “Tipo de servicio”, “Proyecto”, “Lugar de suministro” y “Aperturar Factura Desde”, elegir y digitar la información para el nuevo servicio. 6. Dar clic en el botón “Guardar” y este se mostrará en la lista de servicios de agua inmediatamente, ir al paso 1. 7. Para “Editar” un servicio de agua se debe identificar un servicio de agua de la lista de acuerdo al criterio de búsqueda seleccionado. 8. Dar clic en el botón “Acción” el cual desplegara la opción de “Editar” y “Eliminar” dar clic en el ítem “Editar. 9. Se cargan los datos del contribuyente en formulario editar servicio, listo para editar los campos de interés. 10. Dar clic en el botón “Guardar” y este se actualizará en la tabla de servicios de agua inmediatamente. 11. Para “Eliminar” un servicio de agua se debe identificar un servicio de agua de la lista de acuerdo al criterio de búsqueda seleccionado. 12. Dar clic en el botón “Acción” el cual desplegara la opción de “Editar” y “Eliminar”, dar clic en el ítem “Eliminar”. 13. Mostrar mensaje de confirmación con las opciones “Cancelar” y “Confirmar”. 14. Si “Confirmar”, elimina el servicio de agua, ir al paso 1; si “Cancelar” cancela la eliminación, ir al paso 1.
Precondición	Se debe autenticar como tipo de usuario “Administrador” al momento de ingresar al sistema.
Poscondición	Almacenar la hora y fecha de creación y actualización de un servicio de agua en la base de datos.

Tabla 32: Caso de Uso – Administrar Servicios de Agua.

Caso de Uso Archivar Lecturas

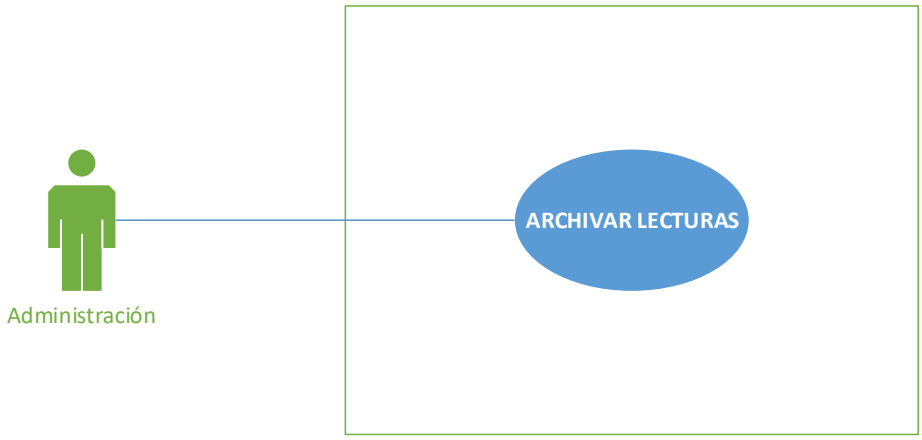


Figura 16: Caso de Uso – Archivar Lecturas.

Caso de uso	#7
Nombre de caso de uso:	ARCHIVAR LECTURAS
Actor Principal	Administración
Dependencia	Acceder al sistema
Descripción	Ingreso de los datos referentes a la lectura de los medidores de agua potable del periodo de pago actual en los proyectos que tienen servicios tipo consumo.
Activar evento	El usuario tipo administrador debe dar clic en la sección "Periodos de pago" del panel de la vista de administración.
Secuencia Principal	1. Mostrar lista de proyectos si existen, con un formulario el lado derecho para agregar "Nueva Fecha" de vencimiento a proyectos, ir al caso "Administrar Periodos de pago", para agregar proyectos ir al caso de uso "Administrar proyectos". 2. Para "Archivar lecturas" se debe identificar un proyecto de la lista con servicios tipo medidor, de acuerdo al criterio de búsqueda seleccionado. 3. Dar clic en el botón "Acción" el cual desplegara la opción de "Editar", "Generar boleta" o "Archivar Lecturas", dar clic en el ítem "Archivar lecturas"; si "Editar", ir al caso

	"Administrar Periodos de pago", si "Generar boleta", ir al caso "Generar boletas de cobro". 4. Se carga el listado de servicios de tipo consumo para el proyecto seleccionado, listo para editar la lectura "Anterior" y "Actual" para cada servicio de ese proyecto. 5. Dar clic en el botón "Guardar todo" y se actualizarán las lecturas inmediatamente y activando el ítem "Generar boletas" para el proyecto seleccionado en la lista fechas de vencimiento. 6. Para "Salir" ir al caso "Acceder al Sistema".
Precondición	Se debe autenticar como tipo de usuario "Administrador" al momento de ingresar al sistema.
Poscondición	Almacenar las lecturas, como también la hora y fecha de creación y actualización de las lecturas de un servicio de agua tipo consumo en la base de datos.

Tabla 33: Caso de Uso – Archivar Lecturas.

Caso de Uso Administrar Periodos de Pago

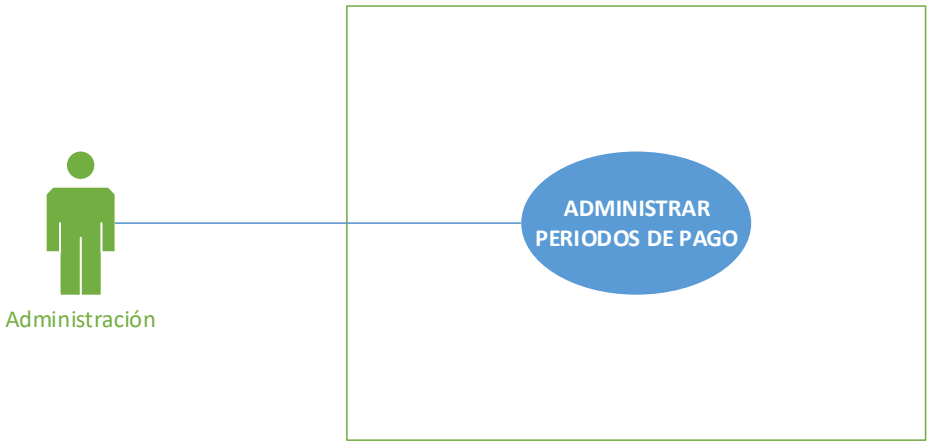


Figura 17: Caso de Uso – Administrar Periodos de Pago.

Caso de uso	#8
Nombre de caso de uso:	ADMINISTRAR PERIODOS DE PAGO.
Actor Principal	Administración
Dependencia	Acceder al Sistema
Descripción	Ingreso de los las fechas de vencimiento de pago para cada proyecto correspondientes al periodo actual y administrar las lecturas de los periodos correspondientes.
Activar evento	El usuario administrador debe dar clic en la sección “Periodos de Pago” del panel de la vista de administración.
Secuencia Principal	1. Mostrar lista de proyectos si existe, con un formulario el lado derecho para agregar “Nueva Fecha” de vencimiento a proyectos, para agregar proyectos ir al caso de uso “Administrar proyectos”. 2. Si “Nueva Fecha” de vencimiento, saltar a paso 3, si “Editar” ir al paso 7, “Salir” ir al paso 13. 3. Para agregar una “Nueva Fecha” de vencimiento, identificar un proyecto de la lista desplegable del formulario. 4. Elegir la “Fecha de vencimiento” de cobro del calendario desplegado para el proyecto elegido. 5. Dar clic en el botón “Guardar” y este se mostrará en la lista de fechas de vencimiento, ir al paso 1. 6. Para “Editar” una fecha de vencimiento se debe identificar un proyecto de la lista de acuerdo al criterio de búsqueda seleccionado. 7. Dar clic en el botón “Acción” el cual desplegara la opción de “Editar”, “Generar boleta” o “Archivar Lecturas”, dar clic en el ítem “Editar”; si “Generar boleta”, ir al caso “Generar boletas de cobro”; si “Archivar Lecturas” ir al caso “Archivar lecturas”. 8. Se cargan el nombre del proyecto bloqueado en formulario editar fecha, listo para editar la “Fecha de vencimiento”. 9. Dar clic en el botón “Guardar” y este se actualizará en la

	tabla de fechas de vencimiento inmediatamente.
Precondición	Se debe autenticar como tipo de usuario "Administrador" al momento de ingresar al sistema.
Poscondición	Almacenar la hora y fecha de creación y actualización de una fecha de vencimiento por proyecto en la base de datos.

Tabla 34: Caso de Uso – Administrar Fechas de Vencimiento.

Caso de Uso Generar Boletas de Cobro

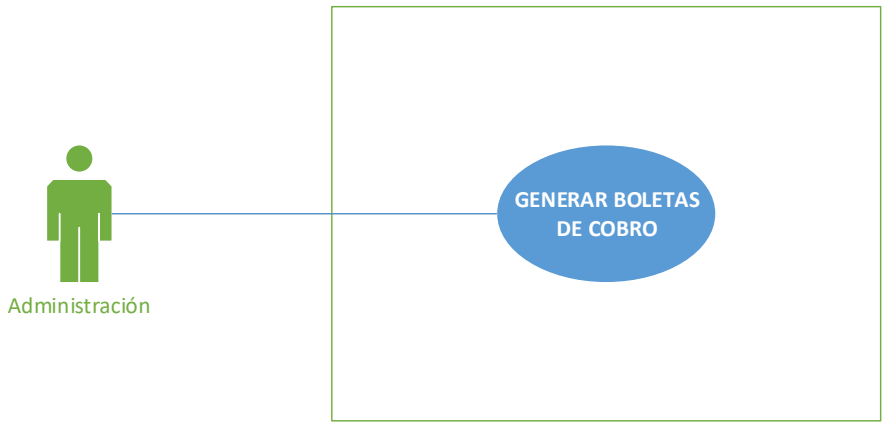


Figura 18: Caso de Uso – Generar Boletas de Cobro.

Caso de uso	#9
Nombre de caso de uso:	GENERAR BOLETAS DE COBRO.
Actor Principal	Administración
Dependencia	Acceder al sistema
Descripción	Generación de las boletas de cobro para cada proyecto de acuerdo a tipo de servicio de agua correspondientes al periodo actual.
Activar evento	El usuario administrador debe dar clic en la sección "Fechas de vencimiento" del panel de la vista de administración.
Secuencia Principal	1. Mostrar lista de fechas de vencimiento si existen, con un formulario el lado derecho para agregar "Nueva Fecha" de vencimiento a proyectos ir al caso de uso "Administrar Periodos de pago".

	2. Para “Generar boleta” de cobro se debe identificar un proyecto de la lista de acuerdo al criterio de búsqueda seleccionado. 3. Dar clic en el botón “Acción” el cual desplegara la opción de “Editar” y “Generar Boleta”, dar clic en el ítem “Generar boleta”; si “Editar”, ir al caso “Administrar fechas de vencimiento”. 4. Se cargan un documento PDF en una nueva pestaña del navegador con las boletas de cobro para el proyecto seleccionado, listo para ser manipulado por el usuario y generando un listado de contribuyentes con estado de pago “Iniciado” en la sección pagos del panel de la vista de administración, para ir a pagos ir al caso de uso “Consultar pagos”.
Precondición	Se debe autenticar como tipo de usuario “Administrador” al momento de ingresar al sistema.
Poscondición	Se crea registros generados en las boletas en la tabla “payments” con estado de pago 1, listos para ser procesados.

Tabla 35: Caso de Uso – Generar Boletas de Cobro.

Caso de Uso Consultar Pagos

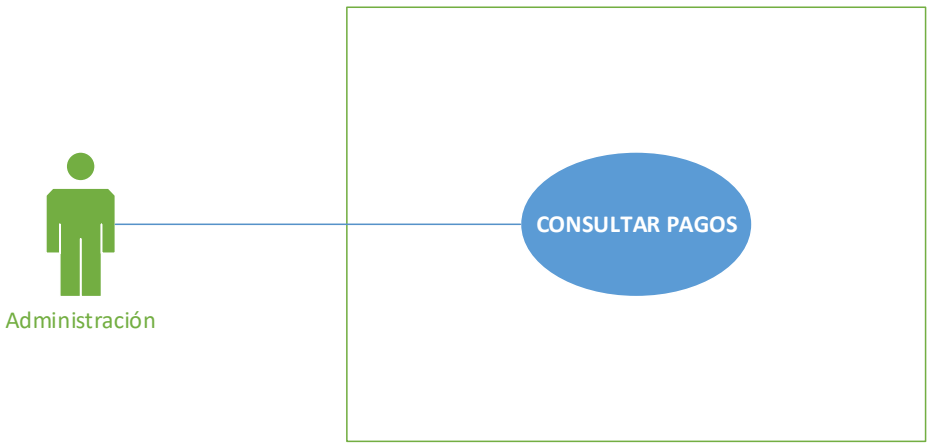


Figura 19: Caso de Uso – Consultar Pagos.

Caso de uso	#10
Nombre de caso de uso:	CONSULTAR PAGOS.
Actor Principal	Administración
Dependencia	Acceder al sistema
Descripción	Consulta de los estados de pago “Iniciado”, “En proceso”, “Pendiente”, y “Finalizado” de los contribuyentes con un servicio de agua potable.
Activar evento	El usuario administrador debe dar clic en la sección “Pagos” del panel de la vista de administración.
Secuencia Principal	1. Mostrar lista de pagos si existen con sus “Nombres”, “Apellidos”, “Tipo de servicio”, “Proyecto” y “Estado de pago” correspondiente para cada contribuyente. 2. Para “Salir” ir al caso “Acceder al Sistema”.
Precondición	Se debe autenticar como tipo de usuario “Administrador” al momento de ingresar al sistema.

Tabla 36: Caso de Uso – Consultar Pagos.

Caso de Uso Efectuar Cobro de Servicio sin Mora

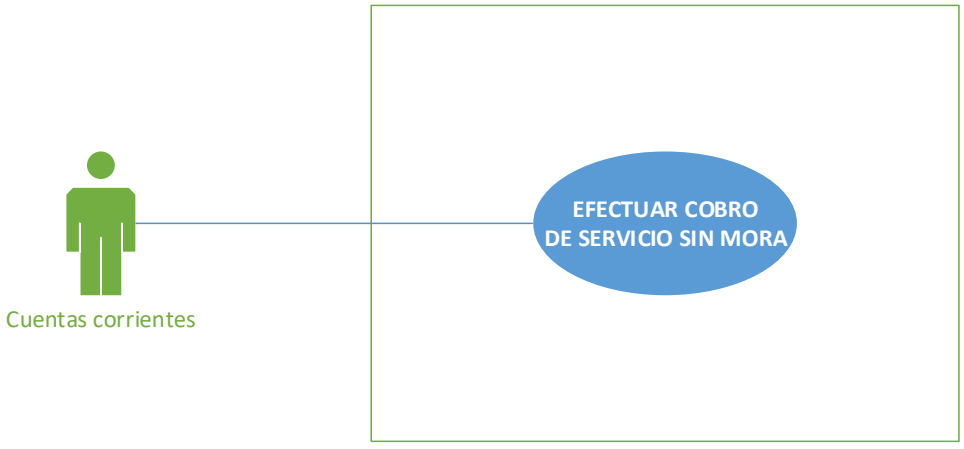


Figura 20: Caso de Uso – Efectuar Cobro de Servicio Sin Mora.

Caso de uso	#11
Nombre de caso de uso:	EFECTUAR COBRO DE SERVICIO SIN MORA.
Actor Principal	Cuentas Corrientes
Dependencia	Acceder al sistema
Descripción	Iniciar el proceso de cobro a un contribuyente con un servicio de agua sin mora.
Activar evento	El usuario Cuentas corrientes debe dar clic en la sección "Servicios de Agua" del panel de la vista de Cuentas Corrientes.
Secuencia Principal	1. Mostrar lista de servicios de agua activos para pagar en el periodo correspondiente si existen. 2. Para "Efectuar cobro" de un servicio se debe identificar un servicio de la lista de acuerdo al criterio de búsqueda seleccionado. 3. Dar clic en el botón "Efectuar cobro" el cual desplegará el formulario "Efectuar cobro" donde se ingresará el código de recibo de pago generado para el contribuyente por un software externo. 4. Dar clic en el botón "Pasar a colecturía" y automáticamente se generará un registro en cola en el módulo colecturía, ir al caso de uso "Despachar servicios en proceso de pago"; se cambiará el estado de pago del contribuyente a "En Proceso" en la lista de pagos del módulo Administrador, ir al caso "Consultar pagos" y se cambiará el estado de pago del contribuyente a "En Proceso" en la lista de servicios del módulo actual. 5. Para "Salir" ir al caso "Acceder al Sistema".
Precondición	Se debe autenticar como tipo de usuario "Cuentas Corrientes" al momento de ingresar al sistema.
Poscondición	Se crea registro en la tabla "payments" con estado de pago 2, listos para ser procesados.

Tabla 37: Caso de Uso – Efectuar Cobro de Servicio Sin Mora.

Caso de Uso Efectuar Cobro de Servicio con Mora

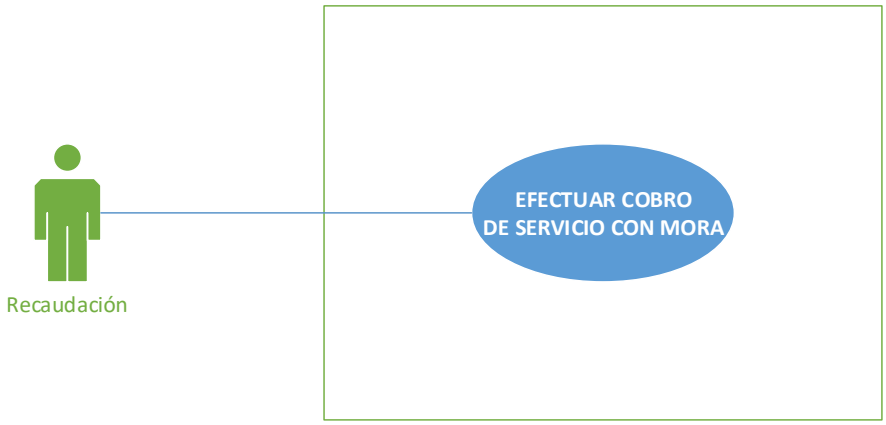


Figura 21: Caso de Uso – Efectuar Cobro de Servicio Con Mora.

Caso de uso	#12
Nombre de caso de uso:	EFECTUAR COBRO DE SERVICIO CON MORA
Actor Principal	Recuperación de mora
Dependencia	Acceder al sistema
Descripción	Iniciar el proceso de cobro a un contribuyente con un servicio de agua con mora.
Activar evento	El tipo de usuario "Recuperación de mora" debe dar clic en la sección "Recuperación de moras" del panel de la vista de Recuperación de mora.
Secuencia Principal	1. Mostrar lista de servicios de agua activos para pagar en el periodo correspondiente si existen. 2. Para "Efectuar cobro" de un servicio se debe identificar un servicio de la lista de acuerdo al criterio de búsqueda seleccionado. 3. Dar clic en el botón "Efectuar cobro" el cual desplegará el formulario "Efectuar cobro" donde se ingresará el código de recibo de pago generado para el contribuyente por un software externo. 4. Dar clic en el botón "Pasar a colecturía" y automáticamente se generará un registro en cola en el módulo colecturía, ir al

	caso de uso “Despachar servicios en proceso de pago”; se cambiará el estado de pago del contribuyente a “En Proceso” en la lista de pagos del módulo Administrador, ir al caso “Consultar pagos” y se cambiará el estado de pago del contribuyente a “En Proceso” en la lista de servicios del módulo actual. 5. Para “Salir” ir al caso “Acceder al Sistema”.
Precondición	Se debe autenticar como tipo de usuario “Recuperación de mora” al momento de ingresar al sistema.
Poscondición	Se crea registro en la tabla “payments” con estado de pago 2, listos para ser procesados.

Tabla 38: Caso de Uso – Efectuar Cobro de Servicio Con Mora.

Caso de Uso Despachar Servicios en Proceso de Pago

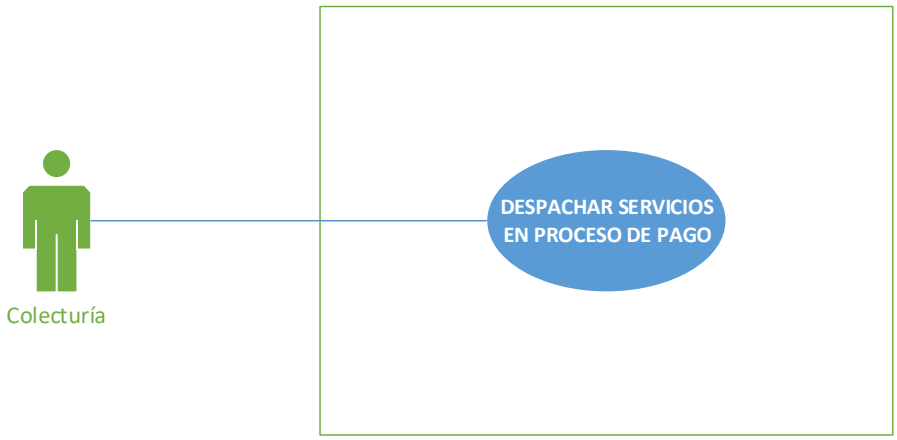


Figura 22: Caso de Uso – Despachar Servicios en Proceso de Pago.

Caso de uso	#13
Nombre de caso de uso:	DESPACHAR SERVICIOS EN PROCESO DE PAGO
Actor Principal	Colecturía.
Dependencia	Acceder al sistema
Descripción	Finalizar el proceso de cobro a un contribuyente con un servicio de agua en proceso de pago.

Activar evento	El tipo de usuario colecturía debe dar clic en la sección "Pagos en cola" del panel de la vista de Colecturía.
Secuencia Principal	1. Mostrar lista de servicios en cola que están en proceso para ser despachados. 2. Para "Despachar" un servicio en cola se debe identificar un servicio de la lista de acuerdo al criterio de búsqueda seleccionado. 3. Dar clic en el botón "Despachar" y automáticamente se eliminará el registro en cola en el módulo actual, se cambiará el estado de pago del contribuyente a "Finalizado" en la lista de pagos del módulo Administrador, ir al caso "Consultar pagos". 4. Para "Salir" ir al caso "Acceder al Sistema".
Precondición	Se debe autenticar como tipo de usuario "Colecturía" al momento de ingresar al sistema.
Poscondición	Se crea registro en la tabla "payments" con estado de pago 4, listos para ser procesados.

Tabla 39: Caso de Uso – Despachar Servicios en Proceso de Pago.

3.3 DISEÑO DE LAS CLASES

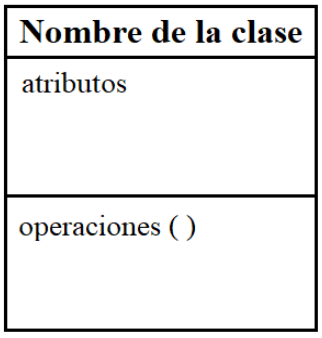
3.3.1 Clases

Una clase es un nivel de abstracción bastante elevado dentro del paradigma de la Programación Orientada a Objetos (POO), ya que sirve principalmente para encapsular atributos y métodos. En UML, una clase representa un objeto o una serie de objetos que poseen características y comportamientos comunes. Estos son representados con rectángulos divididos en 3 particiones.

Primeramente, se coloca el nombre de la clase en la primera partición (centrada, negrita y mayúscula inicial), esta sección es siempre requerida sea que se esté hablando de una clase o un objeto.

Luego se listan los atributos en la segunda partición (alineado a la izquierda, normal y en minúsculas), aquí se describen las cualidades de la clase.

Por último, se escriben las operaciones (en algunos casos llamados métodos) en la tercera partición. Las operaciones describen como la clase interactúa con los datos.



Clase

Figura 23: Diseño de Clases – Clase.

Visibilidad

Se usan marcadores de visibilidad para indicar quienes pueden acceder la información contenida dentro de la clase.

Entre los niveles de acceso (o niveles de visibilidad) más comunes tenemos:

Visibilidad privada. Se denota con un singo '-', esta esconde información a todo lo que está afuera de la partición de la clase.

Visibilidad pública. Se denota con un signo '+', esta permite a las otras clases ver la información que contiene dentro de ella.

Visibilidad protegida. Se denota con un signo '#', esta permite una visibilidad tanto dentro de la clase contenedora como en otras clases fuera de ella, también permite a clases hijas acceder a la información que heredan de una clase padre.

Si no hay declarada ninguna visibilidad, el objeto tendrá **visibilidad de paquete**, es decir, solo se podrá acceder a él desde la propia clase o desde el paquete que lo contiene.

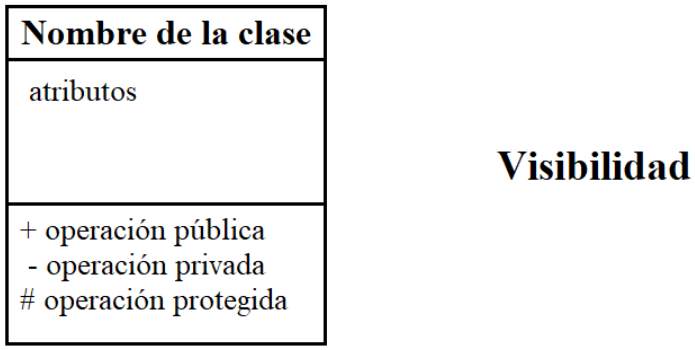


Figura 24: Diseño de Clases – Visibilidad.

3.3.2 Descripción de Clases

A continuación, se describen las clases que conforman el proyecto:

CustomerController	
Descripción: Clase que maneja todas las acciones relacionadas a un contribuyente, y que permite listarlos.	
Atributos	
rowPerPage	Almacena el número de filas de contribuyentes a mostrar por pagina
customers	Almacenara todos los contribuyentes existentes para su respectivo uso.
Métodos	
lastPage()	Método que obtiene la última página del listado de contribuyentes mostrados en su área correspondiente.
currentPage()	Método que devuelve la página actual del listado de contribuyentes si existen.
searchByFirstnameIfNotEmpty()	Permite hacer la búsqueda de contribuyentes por nombre si existen.
isNotEmptySearch()	Método que devuelve la búsqueda si esta no está vacía.
show()	Muestra texto para búsqueda de contribuyentes si el usuario que esta logueado existe.
create()	Método que permite la creación de un nuevo contribuyente y una vez creado actualizar la página.
search()	Permite hacer la búsqueda de un contribuyente.
update()	Identifica a un contribuyente por su Id y permite su actualización mostrándolo en la página.
edit()	Método que hace la edición de un contribuyente a través de su Id y actualizarlo en la respectiva página.
delete()	Permite eliminar un contribuyente a través de su Id y muestra el mensaje respectivo de eliminación.
search()	Método que captura lo que se ha escrito y hace una búsqueda re direccionando a la página actual.
newWaterService()	Método que registra un nuevo servicio de agua a un contribuyente por medio de su Id, mostrando los tipos de apertura, de servicio y zonas actuales.
isNotCheckedInUser()	Verifica que el usuario actual logueado sea un usuario valido y existente para la sesión actual.

Tabla 40: Descripción de Clases – CustomerController.

DefeatOfDatePerZoneController	
Descripción: Clase que permite asignar, y editar fechas de vencimiento para pagos del servicio de agua de los diferentes proyectos existentes.	
Métodos	
show()	Devuelve todas las zonas existentes para la asignación de fechas.
buildLast2DefeatOfDate()	Recorre las zonas por medio de un foreach, obteniendo las dos últimas fechas de vencimiento para así poder determinar el periodo anterior y actual de pago.
create()	Asigna una fecha de vencimiento a una zona determinada con el formato Año/Mes/Día, la activa y la almacena.
isNotValidatesDate()	Recibe una fecha y el id de una zona, determinando una fecha de vencimiento no válida.
checkInDefeatOfDatePerZone()	Método que devuelve una fecha de vencimiento por zona.
pdf()	Recibe las zonas y de acuerdo a las fechas límites de pago para poder generar los recibos de cobro.
storePaymentIfNotCheckIn()	Almacena el pago si no está registrado, usando la última fecha de pago y la fecha de vencimiento actual de la zona.
isApplicableInvoiceWaterService()	Verifica el número de meses que el contribuyente estará obligado a pagar y que este sea igual o mayor a 1.
edit()	Método que permite editar una fecha de vencimiento para un proyecta o zona.
update()	Método que permite actualizar la fecha de vencimiento para una zona,
isNotValidDefeatOfDate()	Verifica que una fecha límite de pago sea válida.
storePayment()	Método privado que recibe el servicio de agua y fecha para almacenar el pago de meses que el contribuyente hará de acuerdo a su servicio de agua y fecha reciente de pago.
updateRecentExcessConsume()	Método que permite guardar y actualizar el exceso reciente para el servicio tipo consumo.
getFirstMonthDate	Método privado que devuelve la fecha correspondiente al primer mes usando la función mktime.
isNotCheckInPayment	Método que recibe la zona y verifica si el pago no está registrado.
getDefeatOfDateActive	Obtiene la fecha de vencimiento activa por zona.

Tabla 41: Descripción de Clases – DefeatOfDatePerZoneController.

HistoryWaterServiceController	
Descripción: Clase que permite mostrar el historial de los servicios de agua adquiridos por un contribuyente creando una observación para cada servicio.	
Métodos	
create()	Método que permite crear una observación del servicio actual y su estado a través del Id de ese servicio.
show()	Permite mostrar el historial de los servicios del contribuyente en la vista correspondiente.

Tabla 42: Descripción de Clases – HistoryWaterServiceController.

LoginController	
Descripción: Clase publica que controla la sesión y el login de un usuario haciendo su verificación respectiva en la base de datos si existe.	
Atributos	
user	Variable privada donde se almacena el usuario que se ha logueado.
Métodos	
login()	Retorna la vista de login.
checkInUser()	Obtiene el nombre de usuario y contraseña encriptada y si este existe lo manda a la página de inicio, caso contrario no hace login.
existsUser()	Método que verifica si el usuario existe y lo retorna.
startSessionAndRedirectToHomePage()	Inicia la sesión y redirige a la página de inicio.
logout()	Destruye la sesión y redirige a la vista raíz.

Tabla 43: Descripción de Clases – LoginController.

PaymentController	
Descripción: Clase que gestiona los pagos de un contribuyente	
Atributos	
rowPerPage	Variable privada para almacenar el número de filas a mostrar en la vista de pagos.
payments	Variable que permite almacenar todos los pagos que se cargan en la vista de pagos.
Métodos	
function_contruct()	Método constructor que carga el número de filas a mostrar en la vista pagos y carga todos los pagos de la base de datos.
create()	Método público que muestra el monto a pagar por el contribuyente en estado de pago 2 (en proceso), es calculado si es un pago normal o si es un adelanto de pago, guardando el pago y el servicio de agua para poder determinarlo y este debe pasar a colecturía.

isExtendedPayment()	Determina si es un pago adelantado donde el mes tiene que ser mayor a 1.
getFirstMonthDate()	Método privado que devuelve la fecha correspondiente al primer día del mes usando la función mktime.
isNotEndsPayment()	Determina si el pago no está finalizado.
show()	Muestra los pagos en proceso retornándolos en la vista de pagos.
end()	Obtiene el Id de pago y pone el estado de pago a 4 (Finalizado).
lastPage()	Devuelve la última página según el número de páginas.
currentPage()	Recibe la página actual de la vista de pagos.
showAll()	Devuelve todos los pagos en la vista de pagos.
onChangeItem()	Asigna el número en la lista de meses para poder hacer pagos adelantados.
search()	Si el tipo de usuario logueado es root, muestra todos los pagos, de otro modo solo los que están en proceso.

Tabla 44: Descripción de Clases – PayemntController.

TypeOfOpeningController	
Descripción: Permite mostrar todos los tipos de apertura que se pueden hacer.	
Métodos	
show()	Método que obtiene todos los tipos de apertura desde la base de datos.

Tabla 45: Descripción de Clases – TypeOfOpeningController.

TypeOfWaterServiceController	
Descripción: Clase que permite la administración de los tipos de servicios, creación, lectura, actualización y eliminación.	
Atributos	
rowsPerPage	Almacena el número de filas de contribuyentes a mostrar por pagina
typesOfWaterService	Almacena los tipos de servicio de agua.
Métodos	
currentPage()	Método que recibe la página actual del listado de tipos de servicio de agua si existen.
show()	Muestra texto para búsqueda de contribuyentes si el usuario que esta logueado existe.
create()	Método que permite la creación de un nuevo tipo de servicio, con su nombre, código y cargo, una vez creado actualizar la página.
delete()	Permite eliminar un tipo de servicio a través de su Id y muestra el mensaje respectivo de eliminación.
edit()	Método que hace la edición de un tipo de servicio a través de su Id y actualizarlo en la respectiva página.
update()	Identifica a un tipo de servicio por su Id y permite su

	actualización mostrándolo en la página.
search()	Permite hacer la búsqueda de un contribuyente, re direccionándolo a la página actual.
isNotEmptyTypeOfWaterService()	Método que devuelve la búsqueda del tipo de servicio si esta no está vacía.
lastPage()	Calcula cual es la última página de acuerdo a los tipos de servicio y las filas por página.

Tabla 46: Descripción de Clases – TypeOfWaterServiceController.

UserController	
Descripción: Clase que administra los usuarios, creación, lectura, actualización y eliminación.	
Métodos	
show()	Obtiene todos los usuarios y tipos de usuarios de la base de datos y los muestra.
create()	Crea un usuario obteniendo los datos ingresados y lanza un mensaje en la sesión y vista actual.
delete()	Elimina un usuario, obteniéndolo por su Id, muestra un mensaje de confirmación y actualiza la vista actual.
edit()	Permite el posicionamiento en el campo y edición de un usuario.
update()	Método que actualiza un usuario obteniendo el id de este.
home()	Vuelve a recalcular el pago para actualizarlo en la base de datos, cuando el usuario se ha logueado.

Tabla 47: Descripción de Clases – UserController.

WaterServiceController	
Descripción: Administra los servicios de agua, permitiendo así mostrarlos.	
Atributos	
rowsPerPage	Almacena el número de registros que van a aparecer por página.
watersService	Almacena los servicios de agua obteniéndolos de la base de datos.
Métodos	
currentPage()	En cuanto a paginación muestra la página actual.
searchByFirstnameIfNotEmpty()	Hace una búsqueda por nombre de contribuyente de acuerdo a servicio de agua.
lastPage()	Obtiene la última página de las que hay existentes.
show()	Muestra los servicios de agua verificando al mismo tiempo si el usuario está registrado.
create()	Crea un nuevo servicio de agua validando si la fecha de

	apertura es correcta, mostrando un mensaje que puede aperturar en el mes actual o a inicios del siguiente. Obtiene todos los datos a usar para aperturar el servicio, como lo es el tipo de servicio, tipo de apertura, zona, pago reciente y lo guarda en la base de datos.
isNotValidatesDate()	Recibe una fecha para validar que a la hora de apertura del servicio de agua se ingrese una fecha válida para el contribuyente.
firstMonthDay()	Obtiene la fecha correspondiente al primer día del mes.
edit()	Método que permite seleccionar el servicio de agua de un contribuyente.
update()	Método que luego de ser validada la fecha de apertura del servicio de agua actualiza el servicio de agua del contribuyente editado y guardando en la base de datos.
delete()	Obtiene el id del servicio de agua y lo elimina de la base.
search()	Método que captura lo que se ha escrito y hace una búsqueda re direccionando a la página actual.
isChangedMonth()	Método que permite capturar si el contribuyente ha extendido el pago este debe ser mayor a 1.
payment()	Recibe el Id del servicio de agua, verificando el usuario logueado previamente mostrando así el monto a pagar dependiendo si es tipo cuota básica o mecha, consumo no se muestra porque este no puede ser extendido.
isWaterServiceCuotaFija()	Recibe un servicio de agua asignándole un 1, que lo identifica como cuota fija.
isWaterServiceMecha()	Recibe un servicio de agua asignándole un 3, que lo identifica como mecha.
readingWaterService()	Recibe un Id de servicio de agua mostrándolo en la página actual de acuerdo a la búsqueda especificada.
isNotCheckedInUser()	Verifica si el usuario está registrado entre los usuarios.

Tabla 48: Descripción de Clases – WaterServiceController.

ValidReadingWaterServiceController	
Descripción: Muestra la vista de todas las lecturas por zona.	
Atributos	
zone	Atributo privado que almacena una zona.
Métodos	
create()	Método que permite mostrar los servicios de agua obteniéndolos por zona y así poder ingresarlas.
upadte()	Método que recibe la lectura de una zona y permite su actualización.
isActionUpdate()	Permite verificar si la acción a realizarse será la de actualización, determinando los requisitos necesarios para llevar a cabo la acción.
store()	Método que recibe la lectura para almacenarla en la base de datos.

Tabla 49: Descripción de Clases – ValidReadingWaterServiceController.

WaterServiceDebtedController	
Descripción: Clase que administra los servicios de agua de contribuyentes con mora.	
Métodos	
show()	Muestra la vista de contribuyentes con mora.
getPaymentDebted()	Obtiene los pagos con mora.
makePayment()	Recibe el Id de un pago de acuerdo al contribuyente seleccionando, cargando el pago y retornando la vista con los pagos en mora.
createPayment()	Prepara el pago para un contribuyente con mora
search()	Hace la búsqueda de los contribuyentes con mora.

Tabla 50: Descripción de Clases – WaterServiceDebtedController.

WaterServiceTypeConsumeController	
Descripción: Administra el tipo de servicio de agua consumo.	
Métodos	
create()	Crea una lectura para el contribuyente seleccionado.
show()	Muestra los servicios de agua de tipo consumo en la vista waterServiceTypeConsume, recibiendo el Id del servicio de agua.
edit()	Método que permite seleccionar el servicio de agua tipo consumo de un contribuyente.
update()	Permite la actualización de una lectura actual o anterior del contribuyente si está activo, mostrando confirmación si la actualización fue exitosa.

Tabla 51: Descripción de Clases – WaterServiceTypeConsumerController.

ZoneController	
Descripción: Clase publica que permite la creación y administración de zonas	
Atributos	
rowsPerPage	Guarda el número de registros de zonas que se mostraran.
zones	Atributo privado que almacena una zona.
Métodos	
construct()	Método constructo que carga las filas por página para mostrar y obtiene todas las zonas.
lastPage()	En paginación obtiene la última página si hay un determinado número de páginas.
show()	Muestra la página actual de la vista de zonas.
create()	Método que crea una zona, guardándola en la base de datos y arrojando un mensaje de creación, redirigiendo la página actual.

delete()	Obtiene el id de la zona y la elimina, mostrando un mensaje de confirmación.
edit()	Permite el posicionamiento en el campo y edición de una determinada zona.
update()	Método que obtiene el id de una zona seleccionada y actualiza en la base de datos de acuerdo al texto ingresado en el campo de zona.
search()	Hace la búsqueda de una zona de acuerdo al texto ingresado en el campo zona.
isNotEmptyZone()	Verifica si el usuario no busco nada en la sesión actual.
currentPage()	Devuelve la página actual si la búsqueda no está vacía retornando la vista zona.
searchByDescriptionIfNotEmpty()	Verifica si la búsqueda para una zona no está vacía para la sesión actual.

Tabla 52: Descripción de Clases – ZoneController.

Customer	
Descripción: Clase que extiende la clase Model que vienen con funcionalidades para solicitar datos del contribuyente que estén en la base de datos.	
Métodos	
waterServices()	Método que sirve para relacionar la tabla customer y waterServices, especificando que un contribuyente puede tener muchos servicios de agua.

Tabla 53: Descripción de Clases – Customer.

DefeatOfDatePerZone	
Descripción: Clase que extiende la clase Model que vienen con funcionalidades para solicitar datos de las zonas que estén en la base de datos.	
Métodos	
zone()	Método público que sirve para relacionar la tabla DefeatOfDatePerZone y zone, especificando que una fecha límite de pago pertenece a una zona.

Tabla 54: Descripción de Clases – DefeatOfDatePerZone.

HistoryWaterService	
Descripción: Clase que extiende la clase Model que vienen con funcionalidades para solicitar datos del historial de servicios de agua para un contribuyente que estén en la base de datos.	
Métodos	
statusWaterService	Método público que sirve para relacionar la tabla statusWaterServices y historyWaterServices, especificando que un estado de servicio pertenece a un historial de servicio de agua.

Tabla 55: Descripción de Clases – HistoryWaterService.

MessageBox	
Descripción: Clase que sirve para administrar los mensajes del sistema.	
Atributos	
type	Variable de tipo privada que almacena el tipo de mensaje.
description	Variable privada que almacena la descripción del mensaje.
icon	Variable privada que almacena el icono del tipo de mensaje a manejar.
className	Variable privada que almacena el nombre de la clase del mensaje a eliminar.
Métodos	
make()	Recibe el tipo de mensaje y descripción para su respectiva creación.
output()	Verifica que tipo de mensaje es, y de acuerdo a eso mostrar si es un mensaje de advertencia o de éxito obteniendo el icono y nombre de clase.
isTypeSuccess()	Asigna valor de uno para indicar que se trata de un mensaje de éxito.
isTypeError()	Asigna valor de dos para indicar que se trata de un mensaje de advertencia.
setIconAndClassName()	Recibe el icono y nombre de la clase a manejar para desplegar los mensajes.

Tabla 56: Descripción de Clases – MessageBox.

PageRendering	
Descripción: Clase que maneja las respuestas que vienen desde el servidor y encapsula los datos.	
Atributos	
urlParser	Captura el tipo de petición HTML que estoy haciendo.
Métodos	
isActionEditAndId()	Método público que recibe el id si se ha editado y devuelve la URL con el id.
isActionNewWaterServiceAndId()	Recibe el id si se ha creado un servicio de agua, retornando la URL de servicio nuevo.
isActionConfirmDeleteAndId()	Retorna la URL del registro porque la acción es de eliminar, devolviendo también el id de este.
isActionEdit()	Recibe la fecha límite de pago por zona y si la respuesta que viene del servidor es de editar.
isActionChangeStatusServiceAndId()	Se activa este método si la petición que hace el cliente es de cambio de estado del servicio de agua, de ser así el servidor devuelve la acción de cambio de estado y abriendo un historial de servicio para el contribuyente.
showAmount()	Recibe un servicio de agua para mostrar el monto a pagar por el contribuyente con dos decimales dependiendo si el estado de este es diferente a finalizado.
isActionPaymentAndId()	En la petición de un pago, el servidor devuelve el id.
isActionPaymentDebtAndId()	Retorna un pago si la petición es de un pago con mora.
isActionReadingAndTypeWaterIdAndWaterService()	Se retorna un servicio de agua de un contribuyente.
isWaterServiceTypeConsume()	Retorna si el tipo de servicio solicitado fue de consumo.
assertEquals()	Hace una comparación devolviendo algo esperado y actual.
isNotFirstPage()	Si no es la primera página devuelve la página actual.
isFirstPage()	Recibe la página actual determinando si es la primera.

isLastPage()	Recibe la página actual y la última determinando si es la última página.
isNotLastPage()	Retorna la página anterior si no es la última.
isLevelRoot()	Si el tipo de usuario es de nivel root, devuelve la sesión como root.
isLevelTreasurer()	Si el usuario logueado es de tipo colecturía devuelve la sesión como tal.
isLevelBankAccount()	Si el usuario logueado es de tipo cuentas corrientes devuelve la sesión como tal.
isLevelRecauda()	Si el usuario logueado es de tipo recaudación devuelve la sesión como tal.
dateFormat()	Retorna el formato de fecha especificado.
isApplicableExtendPayment()	Devuelve si la solicitud es aplicable a un pago extendido.
isCheckInCode()	Retorna código diferente de 0 si el código si la solicitud es un código registrado.
canMakePayment()	Si la solicitud es de hacer pago devuelve la respuesta para un pago sin mora con la sesión de cuentas corrientes.
canTypeReading()	Recibe una fecha de vencimiento por zona si la solicitud es para archivar lectura, verificando que esta tenga una fecha de vencimiento activa y devuelve la respuesta para archivar una lectura en la sesión de administrador.
canMakePaymentDebt()	Si la solicitud es de hacer pago con mora devuelve la respuesta para un pago con mora con la sesión de recaudación.
isPaymentInProceeding()	Devuelve la solicitud a un pago sin mora cuando está en estado 2 (en proceso) y la sesión de usuario de cuentas corrientes.
isPaymentDebtInProceeding()	Devuelve la solicitud a un pago sin mora cuando está en estado 2 (en proceso) y la sesión de usuario de recaudación.

makeMonthsFrom1To()	Devuelve la solicitud a el número de meses para el cálculo de que mes debe pagar son mora o con mora.
isNotRootActive()	Si el usuario root no está activo devuelve una sesión diferente.
canGeneratePdfInZone()	Devuelve la respuesta a la solicitud para generar las boletas de cobro cuando la fecha de vencimiento por zona está activa.
hasWaterServiceTypeConsume()	Recibe una fecha de vencimiento para determinar que el servicio de agua a tratar es de tipo consumo.
canEditDefeatOfDate()	Si la solicitud es de actualizar fecha de vencimiento devuelve la respuesta para la actualización de la fecha de vencimiento de una zona, recibiendo una fecha de vencimiento.
typedReadingWater()	Recibe una fecha de vencimiento si la solicitud es de archivar una lectura de agua tipo consumo y devuelve la respuesta para el ingreso de la lectura.
getCurrentReading()	Devuelve la respuesta a la solicitud para obtener la lectura actual de tipo de servicio consumo.
getDescriptionZone()	Método que obtiene la zona para ser utilizada en otros ámbitos.

Tabla 57: Descripción de Clases – PageRendering.

Payment	
Descripción: Clase que extiende la clase Model que vienen con funcionalidades para solicitar datos de los pagos que estén en la base de datos.	
Métodos	
getWaterService()	Método público que sirve para relacionar la tabla payment y waterService, especificando que un pago pertenece a un servicio de agua.
paymentStatus()	Método público que sirve para relacionar la tabla payment y paymentStatus, especificando que un pago pertenece a un estado de pago.

Tabla 58: Descripción de Clases – Payment.

PaymentCalculator	
Descripción: Clase que permite manejar los cálculos adecuados para los pagos de servicio de agua.	
Atributos	
waterService	Variable privada que almacena el servicio de agua
consumeExcess	Variable privada que almacena el exceso de consumo.
interval	Variable privada que almacena el intervalo de meses a pagar.
amount	Variable privada que almacena la cantidad en dólares a pagar por el servicio.
month	Variable privada que almacena el mes actual.
recentExcess	Variable privada que almacena el exceso de consumo.
Métodos	
makePayment()	Método que recibe un servicio de agua y dependiendo del servicio de agua y si esta al día o no devuelve el pago.
calculatePaymentDependentWaterService()	Método que verifica el pago de acuerdo al tipo de servicio, cuota fija, consumo o mecha.
isWaterServiceBasicBonus ()	Devuelve el tipo de servicio a cuota fija con un id de 1.
isWaterServiceTypeConsume()	Devuelve el tipo de servicio a cuota fija con un id de 2.
calculatePaymentBasicBonus ()	Calcula el pago de cuota fija verificando si este tiene mora, de ser así se emplea una formula preestablecida para sumarle el interés y multa al monto base o si está al día hace el cálculo normal, dependiendo del número de meses.
isApplicableForfeitBasicBonus()	Devuelve la multa fija de $2.86 si la mora es igual o menor a 7 meses.
calculateInterval()	Calcula el mes o intervalo de meses que el contribuyente deberá pagar.
calculatePaymentConsume()	Calcula el pago para el tipo de servicio de consumo, obteniendo el método que calcula el exceso si lo hay y obteniendo el intervalo de meses a cobrar, también en un condicional se determina si no tiene mora y si la tiene sumar la respectiva multa e interés.
isBeforeCriticalMonthIncluded()	Método que permite determinar el cálculo de la mora si es un mes de deuda, es decir calculo normal.
isLeftInCriticalMonth()	Método que permite determinar el cálculo de la mora si son 2 meses y aun no se ha llegado

	la fecha de vencimiento, es decir sumatoria de los 2 meses.
isRightInCriticalMonth()	Método que permite determinar el cálculo de la mora si son 2 meses y después de la fecha de vencimiento, es decir cálculo de mora por 2 meses.
isAfterCriticalMonth()	Método que permite determinar el cálculo de la mora para moras mayores a dos meses.
getAmountPaymentPrev()	Método que retorna la cantidad de pago correspondiente al mes previo.
isNotConsumedWaterService()	Retorna 0 para indicar que no hay exceso en el tipo de servicio consumo.
calculateConsumeExcess()	Calcula el exceso para el tipo de servicio consumo, guardando el valor absoluto de la lectura previa y actual en la variable consumeExcess previamente verificando si este es mayor a 20.
isApplicableExcess()	Método que compara y devuelve si la variable consumeExcess es mayor a 20.
calculatePaymentMecha()	Hace el cálculo para el tipo servicio mecha, determinando si está al día o si esta en mora usar el método para calcular intervalo de meses y aplicar la multa e interés ya preestablecidos.
isApplicableForfeitBasicMecha()	Retorna el intervalo de meses a pagar que es menor o igual a 28, el cual servirá para determinar que la multa base es de $2.86

Tabla 59: Descripción de Clases – PaymentCalculator.

PaymentStatus	
Descripción: Clase que extiende la clase Model que vienen con funcionalidades para solicitar datos de los estados de pago que estén en la base de datos.	
Métodos	
payments()	Método público que sirve para relacionar la tabla paymentStatuses y payments, especificando que un estado de pago puede tener muchos pagos asociados.

Tabla 60: Descripción de Clases – PaymentStatus.

PaymentTicketGenerator	
Descripción: Clase que contiene los métodos esenciales para poder generar las boletas de pago para cada zona.	
Atributos	
zone	Variable de tipo privada que almacena las zonas para la generación de boletas.
interval	Variable de tipo privada que almacena el intervalo de meses a pagar si los hay.
Métodos	
generate()	Recibe una zona y verifica si esta zona tiene asignado un intervalo para pagos iguales o mayores a un mes, generando así la tabla con los servicios de la zona correspondiente.
calculateInterval()	Calcula el intervalo de meses que se usara para aplicar el pago a los contribuyentes, usando la fecha reciente de pago y la fecha límite de pago para hacer el cálculo.
isApplicableInvoiceWaterService()	Verifica el número de meses que el contribuyente estará obligado a pagar y que este sea igual o mayor a 1.
generateTableHtml()	Recibe los servicios de agua y contiene el código HTML y CSS para la generación de la boleta de cobro que le llegara a cada contribuyente de cada zona.
isWaterServiceTypeConsume()	Método que recibe un servicio de agua, retornándolo con un id igual a 2, para identificarlo como tipo consumo.
getDefeatOfDateActive()	Método que devuelve que la fecha límite de pago este activa para la generación de boletas de la zona seleccionada.

Tabla 61: Descripción de Clases – PaymentTicketGenerator.

StatusPaymentPerZoneChanger	
Descripción: Clase que maneja los cambios de estados de pago por zona	
Atributos	
paymentsExpired	Variable privada que almacena un pago expirado.
Métodos	
changePaymentExpiredToPending()	Método que cambia de estado de pago expirado a pendiente.
findPaymentsExpiredPerZoneActive()	Método que obtiene todos los estados de pago para las zonas que tienen una fecha límite de pago activa y así poderlos catalogar como pendientes.
addPaymentExpired()	Recibe una fecha límite de pago y así obtener todos los pagos con id 1, Iniciado y poder pasarlos a pendientes.
putStatusPending()	Obtiene todos los estados de pago que están iniciados

	para pasarlos a pendientes con estado 3, y así guardarlos.
isExpiredDefeatOfDate()	Recibe una fecha límite de pago retornando si la fecha límite de pago es menor a la fecha actual, para poder determinar que la fecha límite expiró.

Tabla 62: Descripción de Clases – StatusPaymentPerZoneChanger.

StatusWaterService	
Descripción: Clase que extiende la clase Model que vienen con funcionalidades para solicitar datos de los estados de servicio de agua que estén en la base de datos.	
Métodos	
historyWaterService()	Método público que sirve para relacionar la tabla statusWaterServices y historyWaterServices, especificando que un historial de servicio puede tener muchos estados de servicio de agua.

Tabla 63: Descripción de Clases – StatusWaterService.

TypeOfOpening	
Descripción: Clase que extiende la clase Model que vienen con funcionalidades para solicitar datos de los tipos de apertura que estén en la base de datos.	
Métodos	
waterServices()	Método público que sirve para relacionar la tabla typeOfOpenings y waterServices, especificando que un tipo de apertura de agua puede tener muchos servicios de agua.

Tabla 64: Descripción de Clases – TypeOfOpening.

TypeOfWaterService	
Descripción: Clase que extiende la clase Model que vienen con funcionalidades para solicitar datos de los tipos de servicio de agua que estén en la base de datos.	
Métodos	
waterServices()	Método público que sirve para relacionar la tabla typeOfWaterServices y waterServices, especificando que un tipo de servicio de agua puede tener muchos servicios de agua asociados.

Tabla 65: Descripción de Clases – TypeOfWaterService.

UrlParser	
Descripción: Clase usada por la clase RenderingPage para obtener las respuestas a las solicitudes que se envían al servidor y devolver las acciones requeridas.	
Atributos	
action	Variable privada que almacena la acción para una determinada solicitud.
id	Variable privada que almacena el id para una determinada acción que se requiera.
Métodos	
getAction()	Obtiene la acción de la solicitud del cliente.
getId()	Obtiene el id dependiendo de la solicitud del cliente.
parse()	Hace el parseo de la URL dependiendo del id y la acción de la solicitud.

Tabla 66: Descripción de Clases – UrlParser.

User	
Descripción: Clase que extiende la clase Model que vienen con funcionalidades para solicitar datos de los usuarios que estén en la base de datos.	
Métodos	
getTypeUser()	Método público que sirve para relacionarla tabla typeUsers y users, y obtener el tipo de usuario especificando que un tipo de usuario pertenece a un usuario.

Tabla 67: Descripción de Clases – User.

WaterService	
Descripción: Clase que extiende la clase Model que vienen con funcionalidades para solicitar datos de los servicios de agua que estén en la base de datos.	
Métodos	
zone()	Método público que sirve para relacionar la tabla waterServices y zones, especificando que un servicio de agua pertenece a una zona.
typeWaterService()	Método público que sirve para relacionar la tabla waterServices y typeOfWaterServices, especificando que un servicio de agua pertenece a un tipo de servicio de agua.
typeOfOpening()	Método público que sirve para relacionar la tabla waterServices y typeOfOpenings, especificando que un servicio de agua pertenece a un tipo de apertura.
customer()	Método público que sirve para relacionar la tabla waterServices y customers, especificando que un servicio de agua pertenece a un contribuyente.
payments()	Método público que sirve para relacionar la tabla

	waterServices y payments, especificando que un servicio de agua puede tener muchos pagos.
waterServiceTypeConsume()	Método público que sirve para relacionar la tabla waterServices y waterServiceTypeConsumes, especificando que un servicio de agua puede tener muchos servicios de tipo consumo o viceversa.

Tabla 68: Descripción de Clases – WaterService.

WaterServiceTypeConsume	
Descripción: Clase que extiende la clase Model que vienen con funcionalidades para solicitar datos de los servicios de agua de tipo consumo que estén en la base de datos.	
Métodos	
waterService()	Método público que sirve para relacionar la tabla waterServices y waterServiceTypeConsumes, especificando que un servicio de agua pertenece a un servicio de tipo consumo.

Tabla 69: Descripción de Clases – WaterServiceTypeConsume.

Zone	
Descripción: Clase que extiende la clase Model que vienen con funcionalidades para solicitar datos de las zonas que estén en la base de datos.	
Métodos	
waterServices()	Método público que sirve para relacionar la tabla waterServices y zones, especificando una zona puede tener muchos servicios de agua.
defeatOfDates()	Método público que sirve para relacionar la tabla defeatOfDatesPerZones y zones, especificando las fechas de vencimiento pueden tener muchas zonas.

Tabla 70: Descripción de Clases – Zone.

3.4 DISEÑO DE INTERFAZ DE USUARIO

3.4.1 Estándares de Diseño de Interfaz

El sistema cuenta con una estandarización de todos sus componentes e interfaces para mantener una presentación consistente y que el usuario se adapte fácilmente a su uso.

Los estándares usados en el diseño visual del sistema se pueden dividir por sus componentes, estos son:

Barra de Navegación

- Barra de navegación en la parte superior color azul oscuro.
- Nombre de usuario activo.
- Logo de la Alcaldía Municipal de Chalchuapa.
- Nombre de sistema.
- Campo de búsqueda y botón buscar.
- Botón “buscar” cambia a color verde al presionar sobre él.
- La barra de navegación siempre está visible.

Figura 25: Estándares de Interfaz– Barra de Navegación Superior.

Panel Menú Izquierdo

- Lista de los distintos módulos del sistema.
- Al pasar el puntero sobre una opción, este se oscurece.

- Cada menú seleccionado cambia a color azul para indicar que es el activo.
- Las letras del menú son de color negro al no estar seleccionadas y cambian a blancas al estar seleccionadas.
- Contenido del menú seleccionado se muestra en el panel de contenido al centro de la página.
- El Panel de Menú siempre está visible.

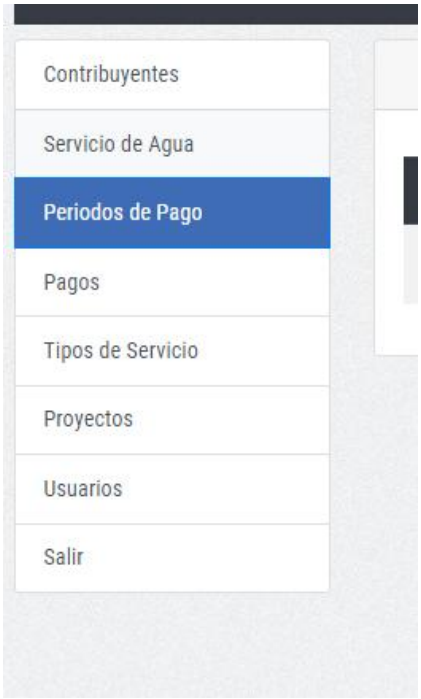


Figura 26: Estándares de Interfaz– Panel Menú Izquierdo

Panel de Contenido

- Ubicado al medio de la página.
- Contiene una tabla mostrando información requerida del modulo.
- Nombre del módulo activo en la parte superior.
- Encabezados de tabla del mismo color que la barra de navegación.

- Texto en los encabezados de las columnas de color blanco.
- Distinción entre un registro y el siguiente.
- Al seleccionar un registro de la tabla este cambia a color esmeralda.
- Botones de Acción en la parte derecha de la tabla.

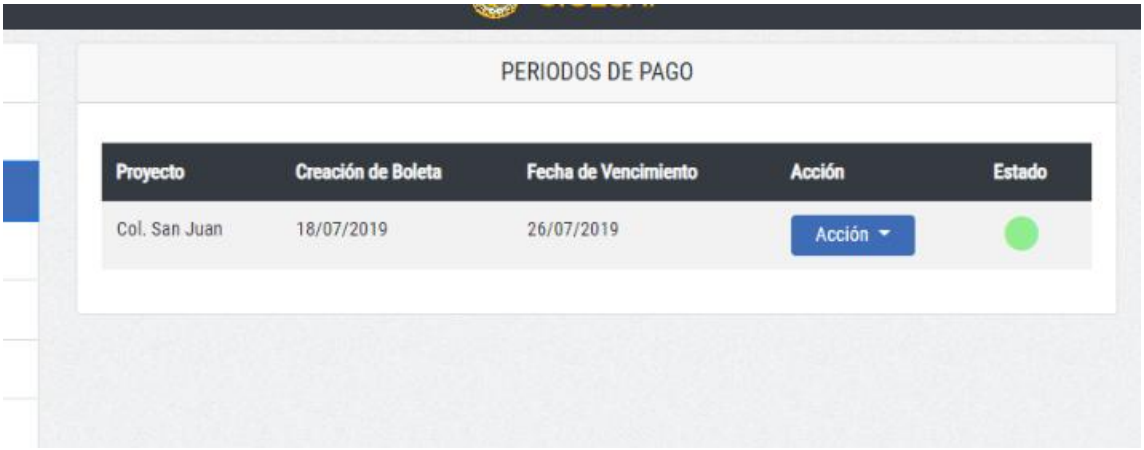


Figura 27: Estándares de Interfaz–Panel de Contenido.

- Paginación de tabla con opción de “Anterior” y “Siguiente”.
- Letras de los botones cambian a azul si existe página siguiente o anterior.

Figura 28: Estándares de Interfaz– Panel de Contenido (Paginación).

Panel de Formularios Derecho

- Panel cambia de acuerdo al modulo seleccionado y a la acción a ejecutar.
- Fondo degradado azul y azul oscuro.
- Nombre del modulo activo.
- Letras de titulo de color blanco.
- Botón de “Guardar” o “Guardar Cambios” de color azul.

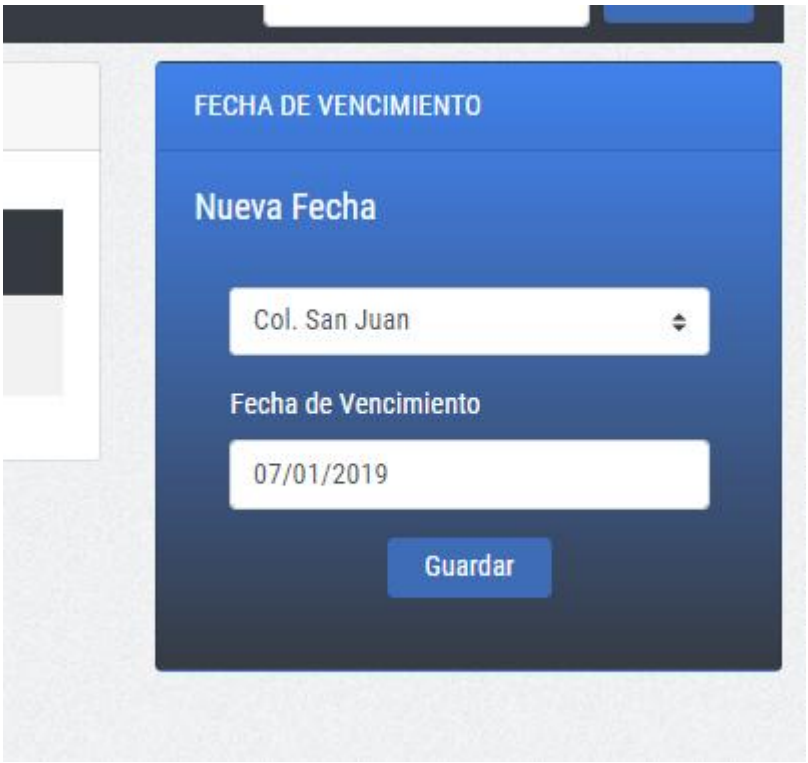


Figura 29: Estándares de Interfaz–Panel de Formularios Derecho.

Botones y Botones de Acción

- Todos los botones y los botones de acción son de color azul oscuro.
- Texto en botones de color blanco.

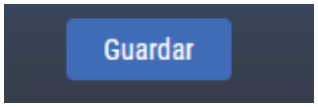


Figura 30: Estándares de Interfaz–Botones.

- Tablasen el Panel de Conteniendo contienen botones de acción según convenga.
- Se despliegan opciones correspondientes al modulo activo al dar clic sobre ellos.
- Dependiendo de la opción seleccionada, así será la respuesta que se mostrará en el Panel de Formulario Derecho.

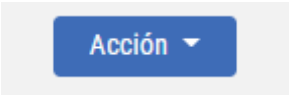


Figura 31: Estándares de Interfaz–Botones de Acción (Inactivo).

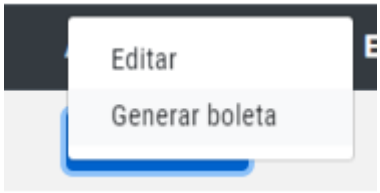


Figura 32: Estándares de Interfaz– Botones de Acción (Activo).

Manejo de Calendarios y Fechas

- Componente calendario en los casos que se requiera el ingreso de fechas.
- Calendario mensual.
- Iconos de atrás y adelante para moverse a través de los distintos meses.
- Muestra la fecha actual de color amarillo.
- Fecha seleccionada de color azul.

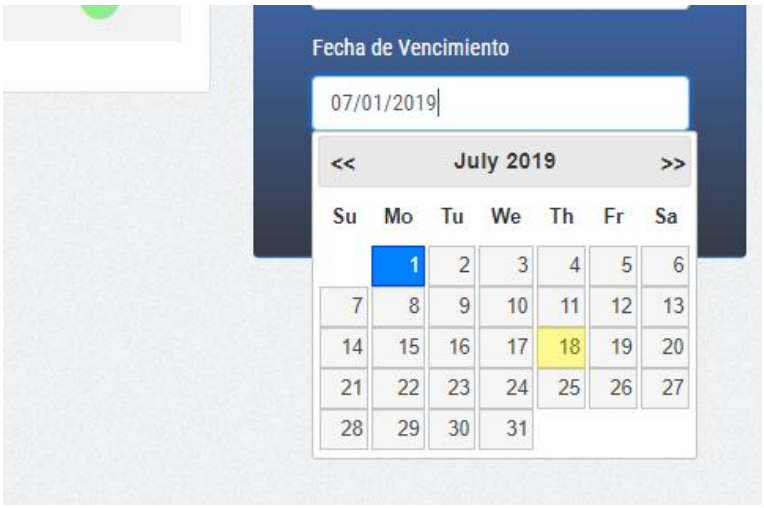


Figura 33: Estándares de Interfaz– Manejo de Calendarios y Fechas.

Ventanas de Confirmación Emergentes

- Ventanas de Confirmación al querer eliminar cualquier registro.
- Muestra en el titulo, el modulo que se está operando.
- Cuenta con botones “Cancelar” y “Confirmar”.
- Fondo se oscurece al aparecer venta de confirmación.

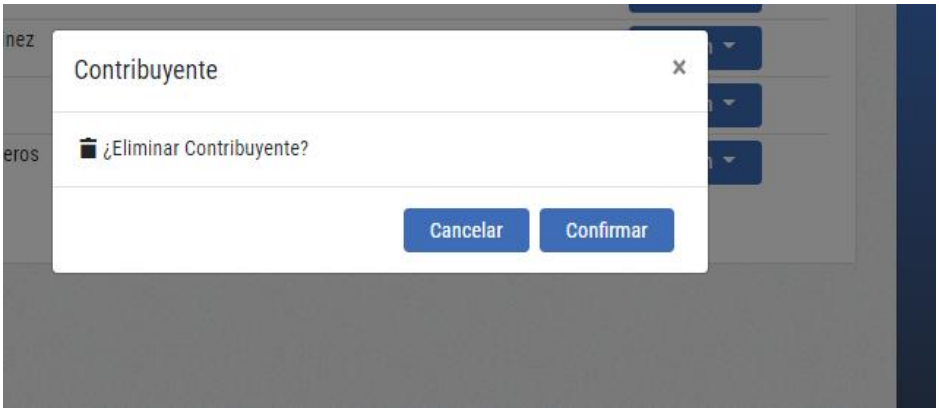


Figura 34: Estándares de Interfaz– Ventanas de Confirmación Emergentes.

3.4.2 Interfaces del Sistema

Al diseñar las interfaces del sistema, uno de los principales era el mantener una vista simplificada y consistente, que facilitara al usuario su navegación por los distintos módulos del sistema. A continuación, se muestran algunas de las pantallas del proyecto informático:

Figura 35: Interfaz de Usuario – Interfaz de Login.

Figura 36: Interfaz de Usuario – Interfaz de Bienvenida.

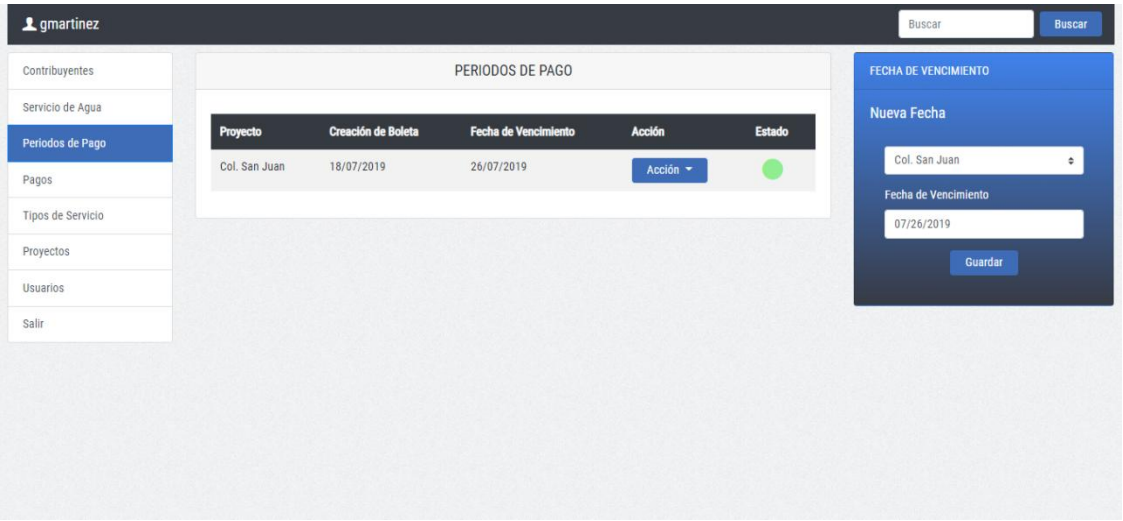


Figura 37: Interfaz de Usuario – Interfaz de Módulo Periodo de Pago.

CAPÍTULO IV

DESARROLLO DEL SISTEMA

CAPÍTULO IV DESARROLLO DEL SISTEMA

4.1 ESTÁNDARES DE DESARROLLO

4.1.1 Desarrollo del Software

El desarrollo de software es en donde se desencadena el diseño de este proyecto. En esta parte de la etapa se describen de forma detallada las herramientas utilizadas para el desarrollo y pruebas, además se especifican los pasos necesarios para el plan de implementación del proyecto.

La planeación inicial de las actividades del proyecto tenía un fin: entregar una herramienta de software funcional y adaptable fácilmente al comportamiento en conjunto de todo el sistema para la Unidad de Administración Tributaria de la Alcaldía Municipal de Chalchuapa. Durante el desarrollo, esta planeación se fue modificando debido a diferentes circunstancias como la adición de nuevas funcionalidades al sistema o la modificación de estas.

Para el desarrollo del proyecto llamado: “SISTEMA INFORMÁTICO PARA LA ADMINISTRACIÓN Y GESTIÓN DE COBRANZAS DEL SERVICIO DE AGUA POTABLE PARA LA UNIDAD DE ADMINISTRACIÓN TRIBUTARIA DE LA ALCALDÍA MUNICIPAL DE CHALCHUAPA.”, aplicación web conocida como “SIGESAP”, se usaron diferentes estándares y tecnologías que se detallan en las secciones de este capítulo.

4.1.2 Metodología del Desarrollo

Utilizar una metodología de desarrollo es de suma importancia, ya que el desarrollo de un programa para la resolución de un problema es una tarea compleja y es necesario tener en cuenta de manera simultánea muchos elementos.

Una metodología de programación es un conjunto de métodos, principios y reglas que permiten enfrentar de manera sistemática el desarrollo de un programa que resuelve una problemática. Estas metodologías generalmente se estructuran como una secuencia de pasos que parten de la definición del problema y culminan con un programa que lo resuelve.

A continuación, se presenta de manera general los pasos en la metodología que se usó para el desarrollo de la aplicación web:

El Diálogo. Para comprender totalmente el problema a resolver.

La Especificación. Con la cual se establece de manera precisa las entradas, salidas y las condiciones que deben cumplir.

Diseño. En esta etapa se construye un algoritmo que cumpla con la especificación.

Codificación. Se traduce el algoritmo a un lenguaje de programación.

Prueba y Verificación. Se realizan pruebas del programa implementado para determinar su validez en la resolución del problema.

La metodología de programación implementada es la programación orientada a objetos, el cual es un paradigma de programación orientado a mejorar la claridad, calidad y tiempo de desarrollo de un programa de computadora.

4.1.3 Estándares de Desarrollo

En este apartado se describen los estándares de desarrollo utilizados, donde se retoman los procesos más comunes como lo son ingreso de registros, gestión y consultas/reportes de la aplicación. De la misma forma se describe el diagrama jerárquico de procesos, el cual se reflejará en los módulos con sus procesos a realizar, estándares de migraciones a la base de datos, y la terminología utilizada en el desarrollo.

Un estándar de desarrollo completo comprende todos los aspectos de la generación de código. Los programadores deben implementar un estándar de forma prudente y además, éste debe tender siempre a lo práctico. Un código fuente completo debe reflejar un estilo uniforme, como si un único programador hubiera escrito todo el código. Al comenzar un proyecto de software, se establece un estándar de codificación para asegurarse de que todos los programadores del proyecto trabajen de forma coordinada.

La ventaja de este tipo de codificación es la facilidad con que el sistema de software puede modificarse para añadirle nuevas características, modificar las ya existentes, depurar errores, o mejorar el rendimiento.

Un estándar de desarrollo permite:

- Definir la escritura y organización del código fuente de un programa.
- Facilita a un programador la modificación del código fuente del programa.
- Definir la forma en que deben ser declaradas las variables, las clases, los comentarios.
- Mejora el análisis del código durante el proceso de desarrollo.

Los estándares de programación establecidos para el sistema informático son los siguientes:

- Programación modular, el cual consiste en dividir el programa en módulos o subprogramas con el fin de hacerlo más legible y manejable. Para ello se desarrolló una página principal que contiene varios módulos dentro de ella (todos los módulos que se necesitan según la especificación de la institución).

- Permitir la fácil modificación y corrección del programa una vez este sea terminado.

- Uso de funciones que se puedan reutilizar, y con ello lograr la disminución de líneas de código.

- Entre las estructuras lógicas de programación estructurada que se usarán se tienen: if-else, for, foreach.

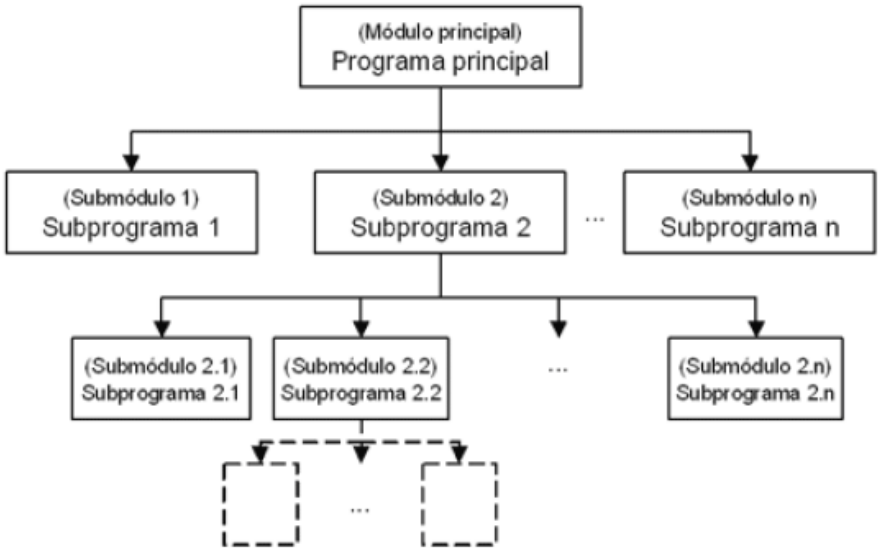


Figura 38: Diagrama de Programación Modular.

4.1.4 Estándares de Base de Datos

Uno de los instrumentos que facilitan la tarea a lo largo del ciclo de vida del desarrollo de un software para asegurar la calidad del mismo, es la adopción de estándares de diseño de bases de datos.

El uso de estos estándares tiene innumerables ventajas, entre algunas de estas ventajas tenemos:

- Asegurar la legibilidad del modelo de datos, inclusive para personas que no están relacionadas con el ambiente informático, en etapas de análisis y diseño.
- Facilitar la portabilidad entre motores de bases de datos, plataformas y aplicaciones.
- Facilitar la tarea de los programadores en el desarrollo de los sistemas.

Es por esto que el diseño de las tablas de las bases de datos a desarrollar de la aplicación cumple ciertos requisitos, detallados en el presente documento. Brevemente se resume en los siguientes puntos:

- Para la creación de las tablas a través de las migraciones se separará por un guion bajo cuando el nombre de las tablas tenga más de una palabra, ejemplo de ello las tablas: water_services o month_collectors.
- La base de datos de la aplicación, se ha denominado bajo el nombre de “sigesap_db” y consta de catorce tablas relacionadas entre sí.

4.1.5 Diagrama Jerárquico de Procesos

El diagrama jerárquico de procesos de la aplicación web, representa de forma gráfica la manera en que los usuarios del sistema pueden realizar las actividades que actualmente realizan de manera manual, de forma mecanizada utilizando las herramientas que la aplicación lleva inmersas para tal efecto.

En la siguiente figura se muestra los módulos principales de la aplicación.

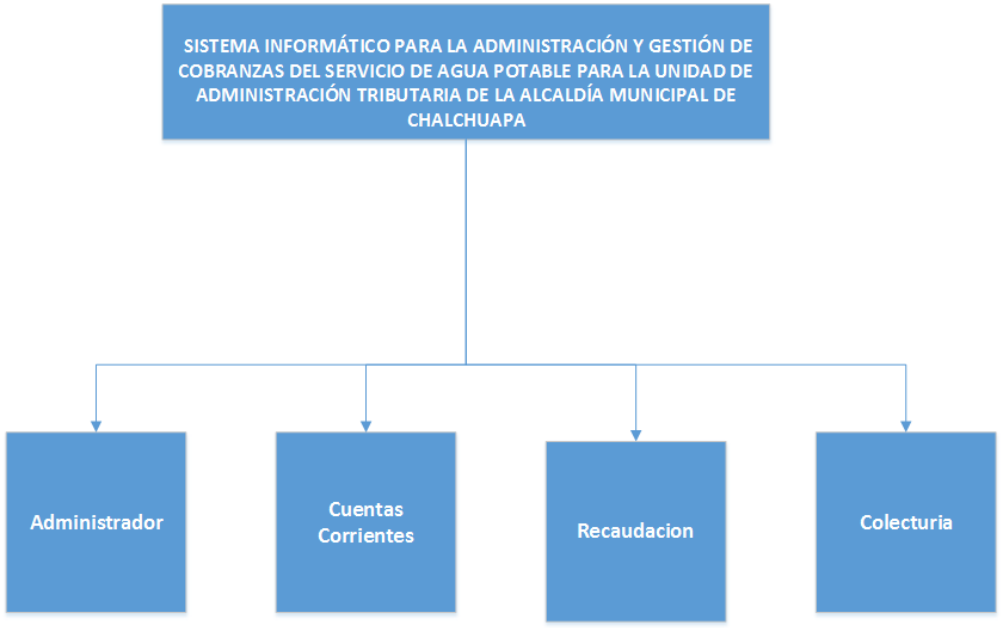


Figura 39: Diagrama de Módulos de la Aplicación.

4.2 TECNOLOGÍAS DE DESARROLLO.

Una vez determinado el tipo de aplicación a desarrollar, se determinarán las tecnologías que le permitan al equipo de desarrollo alcanzar los objetivos fijados al inicio de la investigación.

Se debe tener en cuenta el tipo de tecnologías y herramientas a utilizar, ya que estas pueden infringir los límites establecidos por el personal encargado del área de informática en la Alcaldía Municipal de Chalchuapa.

4.2.1 Leguajes de Programación

En este apartado se definen las tecnologías utilizadas para el desarrollo de la aplicación, a continuación, se detallan cada una de ellas.

PHP 7.3

Lenguaje de programación de uso general de código del lado del servidor, originalmente diseñado para el desarrollo web de contenido dinámico.

Según su página oficial PHP es un lenguaje de código abierto muy popular en el mundo, adecuado para desarrollo web y que puede ser incrustado de manera muy flexible con HTML5. En internet, un gran número de páginas y portales web están usando PHP de manera abierta.

La instalación es realmente sencilla gracias a varios servidores locales como los siguientes: Lampp, Xampp, vertigroServ, WAMPServer, BitName y **Laragon**.

Para el desarrollo del sistema se ha escogido Laragon para su instalación porque ya viene pre instalado y listo para trabajar en plataformas orientadas a la web.

4.2.2 Frameworks

Laravel 5.8.10

Laravel es uno de los frameworks de código abierto más fáciles de asimilar para PHP. Es simple, muy potente y tiene una interfaz elegante y fácil de usar. Fue creado en 2011 y tiene una gran influencia de frameworks como Ruby onRails, Sinatra y ASP.NET MVC[5].

Características:

- ✓ Sistema de ruteo, también RESTful
- ✓ Blade, motor de plantillas
- ✓ Peticiones Fluent
- ✓ EloquentORM
- ✓ Basado en Composer
- ✓ Soporte para el caché
- ✓ Soporte para MVC
- ✓ Usa componentes de Symfony
- ✓ Adopta las especificacionesPSR-2 y PSR-4

[5]¿Que es Laravel? (2015, Diciembre)
Recuperado de:https://www.arsys.es/blog/programacion/que-es-laravel/

Bootstrap 4.3

Bootstrap, es un framework originalmente creado por Twitter, que permite crear interfaces web con CSS y JavaScript. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como extensiones de JavaScript adicionales. A diferencia de muchos frameworks web, solo se ocupa del desarrollo front-end.

Puede descargarse desde el siguiente link en su versión 4.3:

https://getbootstrap.com/docs/4.3/getting-started/download/

4.2.3 Librerías

jQuery

jQuery es una biblioteca de JavaScript rápida, pequeña y con muchas funciones. Hace que la manipulación de documentos HTML, el manejo de eventos, la animación y Ajax sean mucho más simples con una API fácil de usar que funciona en una gran cantidad de navegadores.

Con una combinación de versatilidad y extensibilidad, jQuery ha cambiado la forma en que millones de personas escriben JavaScript.

El enlace de descargar es la siguiente:

https://code.jquery.com/jquery-3.3.1.min.js

4.2.4 Diseño Web

HTML 5

HTML5 (Hyper Text MarkupLanguage) es un lenguaje de marcado de texto usado para estructurar y presentar el contenido de las páginas web. Es uno de los aspectos fundamentales para el funcionamiento de los sitios web.

A fines del año 2014, la W3C recomendó HTML 5 para transformarse en el estándar a ser usado en el desarrollo de proyectos venideros.

CSS 3

CSS significa “Cascade Style Sheets”, también llamado Hojas de Estilo en Cascada. CSS es un lenguaje que se emplea para dar formato a un sitio web. Es decir, funciona en conjunto con los archivos HTML. Por esta razón, es conveniente tener conocimientos tanto de HTML como de CSS.

Durante el desarrollo de este proyecto se usó esta herramienta de diseño para dar estructura a las vistas de los módulos.

4.3 HERRAMIENTAS DE DESARROLLO.

Una vez seleccionado el lenguaje de programación, frameworks y bibliotecas a usar se hace un estudio de las posibles herramientas de trabajo por parte del equipo desarrollador. Llegando a un filtro y por ultimo haciendo una selección tomando en cuenta la calidad, estabilidad, documentación, facilidad de implementación y requisitos de las mismas herramientas.

4.3.1 Entorno de Desarrollo

JetBrains PhpStorm 2019.1.2

Una de las mejores herramientas si se trabaja en un proyecto con PHP (o con el framework Laravel) es PhpStorm. Provee de un editor para PHP, HTML y JavaScript con análisis de código en el momento, prevención de errores y refactoreo automatizado para código PHP y JavaScript.

Las características principales de PhpStorm son:

- Asistencia de código inteligente.
- Navegación de código inteligente.
- Refactorización rápida y segura.
- Fácil debug y test.
- Control de versiones con Git, SVN y Mercurial; con soporte para Vagrant, Docker y Composer.

4.3.2 Servidor Web

Laragon 4.0

Laragon ofrece una forma rápida y sencilla de iniciar un desarrollo de Windows aislado (como una máquina virtual, no toca el sistema operativo).

Los usuarios pueden instalarlo como software, iniciarlo, hacer su programación y simplemente salir cuando haya terminado. La plataforma viene preinstalada con muchas aplicaciones populares como Node.js, PHP, Apache, Composer y HeidiSQL/MySQL.

Enlace donde descargar el software:

https://sourceforge.net/projects/laragon/files/releases/4.0/laragon-full.exe

4.3.3 Gestores de Base de Datos

HeidiSQL 9.5.0

Como gestor de base de datos se usará el que viene preinstalado con Laragon, HeidiSQL. Inicialmente conocido como MYSQL-Front, es un software libre y de código abierto que permite conectarse a servidores MySQL, así como Microsoft SQL Server y PostgreSQL.

MySQL Workbench 8.0

MySQL Workbench es una herramienta gráfica para trabajar con servidores y base de datos de MySQL. Soporta servidores MySQL versión 5.6 ó superiores. Provee a DBAs y desarrolladores un entorno con varias herramientas integradas tales como:

- Modelado y Diseño de Bases de Datos
- Administración de Base de Datos
- Migración de datos

El enlace de descarga de MySQL Workbench es el siguiente:

https://dev.mysql.com/downloads/workbench

4.4 SEGURIDAD DEL SISTEMA

La seguridad en las aplicaciones web es un factor muy importante para el buen funcionamiento de los sistemas informáticos. La seguridad informática se encarga específicamente de la seguridad de sitios web, aplicaciones web y servicios web.

Durante el desarrollo del proyecto, se establecieron reglas y métodos de seguridad a implementar para la aplicación, dado que se manejan datos muy importantes de contribuyentes y estos no deben ser vulnerables.

A un alto nivel, la seguridad de aplicaciones web se basa en los principios de la seguridad de aplicaciones, pero aplicadas específicamente a la World Wide Web. Las aplicaciones, comúnmente son desarrolladas usando lenguajes de programación tales como PHP, Python, Ruby, ASP.NET, JSP, entre otros.

Mientras que la seguridad se basa fundamentalmente en las personas y los procesos, existen varias soluciones técnicas a considerar cuando se diseña, construye y prueban aplicaciones web seguras.

A un alto nivel, estas soluciones incluyen:

- **Escáner de Vulnerabilidad.** Escáner de seguridad de aplicaciones web.
- **Fuzzing.** Herramientas utilizadas para pruebas de entrada.
- **Firewalls de aplicación web (WAF).** Utilizada para brindar protección tipo firewall en la capa de aplicación web.
- **Cracking de contraseñas.** Herramienta de prueba de fuerza de contraseñas.

Durante el desarrollo del sistema se utilizarán las herramientas de fuzzing, una técnica de pruebas de software, que implica proporcionar datos inválidos, inesperados o aleatorios a las entradas de un programa de ordenador.

La técnica de fuzzing se utiliza normalmente para descubrir problemas de seguridad en software o sistemas de ordenadores. Es una forma de testeo aleatorio que se ha usado para comprobar tanto hardware como software.

4.4.1 Autenticación

El concepto de autentificación viene asociado a la comprobación del origen de la información, y de la identidad de los agentes que interactúan con un sistema. En general, y debido a los diferentes escenarios que pueden darse, se distinguen tres tipos de autentificación:

- **Autentificación de mensaje.** Garantiza la procedencia de un mensaje conocido, de forma que podamos asegurarnos de que no es una falsificación. Este proceso es el que se utiliza en las firmas digitales, o en los sistemas de credenciales a través de los cuales ciertos elementos de una red se identifican frente a otros.
- **Autentificación de usuario mediante contraseña.** En este caso se trata de garantizar la presencia física de un usuario legal en algún punto del sistema. Para ello deberá hacer uso de una contraseña, que le permita identificarse.
- **Autentificación de dispositivo.** Se trata de garantizar la presencia frente al sistema de un dispositivo en concreto.

En SIGESAP se usa el tipo de autentificación por token y autenticación de usuario mediante contraseña; cada usuario tiene diferentes roles, por tanto, se les asignara un nombre de usuario y contraseña garantizando así la seguridad del mismo y que no puedan interferir en las tareas de los demás usuarios.

Las sesiones para cada usuario mediante autenticación por tokense detallan a continuación:

El cliente envía los datos de login.

A través de un formulario de login en la aplicación se obtiene el usuario y la clave de acceso. Estos datos se enviarán al servidor. En este punto pueden pasar dos cosas:

- Si el login es correcto, se da al usuario por autenticado. Entonces se genera un token en el lado del servidor, que se enviará al cliente de vuelta.
- Si el login no fue correcto, entonces se le manda un código de error al usuario.

Almacenar el token.

En el caso que se realizase un intento correcto de autenticación, y por tanto el servidor devolviese el token, en el lado del cliente se deberá almacenar el token, para poder usarlo más tarde.

Lo ideal será almacenar el token en el navegador, en un lugar donde se pueda acceder a él en futuras consultas. Lo más normal para almacenar el token sería el LocalStorage del navegador.

Envío del token en posteriores accesos.

Cada vez que el usuario, una vez autenticado, desee acceder de nuevo a un recurso de ese API, entonces tendrá que enviar el token al servidor para informarle que es él mismo y que ya se autenticó correctamente en un paso anterior.

Al recibir el token, el servidor lo tendrá que verificar y, si lo encuentra válido, sabrá que corresponde a un usuario de la aplicación, por lo que podrá conceder el acceso al recurso consultado.

4.4.2 Encriptación

Algoritmo de Reducción Criptográfico de Contraseña o también llamado MD5 es una codificación de 128 bits que es representada típicamente como un número de 32 símbolos hexadecimales. En el sistema es usado para encriptar las contraseñas de los usuarios registrados en la base de datos por medio de la vista de login.

Este se ve reflejado en la clase LoginController en la función checkInUser del proyecto:

```
public function checkInUser(UserFormRequest $request) {
        $user = User::where("name", $request->name)
                ->where("password", md5($request->password))
                ->first();
        $this->user = $user;
        if ($this->existsUser())
                return $this->startSessionAndRedirectToHomePage();
        returnRedirect::to("/");
}
```

4.4.3 Roles

En la aplicación se manejan cuatro módulos, esto conlleva un control adecuado y seguro para el manejo de información. Los usuarios manejaran el tipo de información que les corresponde, siendo los roles del sistema los siguientes:

Rol	Función
Administrador (root)	Puede agregar contribuyentes, asignarles un servicio de agua, modificar, eliminar, puede acceder a las vistas de los contribuyentes que están en cola de pago y sus respectivos estados de pago, pero este usuario no puede despachar un contribuyente ni asignarle pago alguno.
Cuentas Corrientes (cuentas)	Solamente puede buscar contribuyentes y asignarles un pago para pasar a colecturía.
Recuperación de Mora (recaudación)	Este usuario solamente puede buscar contribuyentes y asignarles un pago moroso para pasar a colecturía.
Colecturía (colecturía)	El usuario con este rol, solamente puede despachar a los contribuyentes en cola para realizar un pago evitando las acciones CRUD.

Tabla 71: Roles y Funciones del sistema.

4.4.4 Conexiones Seguras

El protocolo de seguridad SSL, proporciona autenticación y privacidad de la información entre extremos sobre internet mediante el uso de criptografía.

Habitualmente, solo el servidor es autenticado (es decir, se garantiza su identidad) mientras que el cliente se mantiene sin autenticar.

SSL implica una serie de fases básicas:

- Negociar entre las partes el algoritmo que se usará en la comunicación.
- Intercambio de claves públicas y autenticación basada en certificados digitales.
- Cifrado del tráfico basado en cifrado simétrico.

Durante la primera fase, el cliente y el servidor negocian qué algoritmos criptográficos se van a usar.

4.5 PRUEBAS DEL SISTEMA

Algunos de los mayores desafíos de los proyectos de desarrollo de software se pueden solucionar realizando actividades sencillas de pruebas, el objetivo principal de las pruebas de software es proporcionar información objetiva e independiente sobre la calidad del producto.

Las pruebas son básicamente un conjunto de actividades realizadas dentro del desarrollo y estas actividades podrán ser implementadas en cualquier momento de dicho proceso de desarrollo.

Existen algunos tipos de pruebas de software, entre ellas están:

Pruebas Estáticas

Son el tipo de pruebas que se realizan sin ejecutar el código de la aplicación, es decir, se realizan revisiones y análisis estáticos a la documentación del proyecto. Dentro de las revisiones se pueden abordar especificaciones de requerimientos, especificaciones de prueba, guías de usuario, entre otros.

Pruebas Dinámicas

Son todas aquellas pruebas que requieren la ejecución de la aplicación. Las pruebas dinámicas permiten el uso de técnicas de caja negra y caja blanca con mayor amplitud. Debido a la naturaleza dinámica de la ejecución de pruebas es posible medir con mayor precisión el comportamiento de la aplicación desarrollada.

4.5.1 Especificaciones de las Pruebas

Generalmente es posible definir una prueba como la ejecución de un software, un módulo, o un trozo de código, con el objetivo de identificar uno o varios errores, defectos y fallos, éstos errores dan lugar a los fallos que seguramente generarán aspectos negativos por los usuarios que verán al software como un producto no fiable.

Algunos de las características de las pruebas de software son:

- Demostrar al desarrollador y al cliente que el software satisface sus requerimientos.
- Encontrar defectos en el software, que su comportamiento es incorrecto, no es deseable o no cumple con su especificación.
- Una prueba tiene éxito si descubre un defecto.
- Una prueba fracasa si hay defectos, pero no los descubre.

4.5.2 Metodología de las Pruebas

Las pruebas del software son los procesos de ejecución de un programa con el fin de encontrar errores para solucionarlos y así determinar con certeza que los requerimientos de desarrollo de la aplicación son cumplidos sin error alguno, es por ello que las pruebas realizadas en la aplicación web fueron las siguientes:

Pruebas Unitarias

Se concentra en el esfuerzo de verificación de la unidad más pequeña del diseño del software, el componente o módulo del software, se puede aplicar en paralelo a varios componentes.

Pruebas de Integración

Es una técnica sistemática para construir la arquitectura del software, es decir, integrar todas las partes que formarán todo el software, mientras, al mismo tiempo, se aplican las pruebas para descubrir errores asociados con la interfaz.

Pruebas del Sistema

Se utilizan al final del desarrollo el software ya que se incorporan otros elementos del sistema (hardware, personas e información) y se realiza una serie de pruebas que se detallan a continuación:

- **Prueba de Seguridad:** Comprueba que los mecanismos de protección integrados en el sistema realmente lo protejan de irrupciones inapropiadas, durante esta prueba quién la aplica desempeña el papel del individuo que desea entrar en el sistema.

- **Prueba de Resistencia:** está diseñadas para confrontar los programas con situaciones anormales, cuando se aplica la prueba, se tratará de sobrecargar el programa.
- **Prueba de Desempeño:** está diseñada para probar el desempeño del software en tiempo de ejecución dentro del contexto de un sistema integrado, se aplica en todos los pasos del proceso de las pruebas.

Pruebas de Aceptación

Empiezan tras la culminación de la prueba de integración, cuando se han ejercitado los componentes individuales. Se ha terminado de ensamblar el software como paquete y se han descubierto y corregido los errores de interfaz, se concentra en las acciones visibles para el usuario y en la salida del sistema que éste puede reconocer; dicho de otra forma, se satisface las expectativas razonables del cliente.

4.5.3 Pruebas Realizadas

Pruebas Unitarias

Es la prueba de cada pantalla, que se realizó con el objetivo de verificar que la validación de los datos que se ingresarán sea la correcta, al mismo tiempo los procesos que se realizan tenga resultado satisfactorio tanto para el usuario como para el desarrollador.

Se hicieron pruebas de validación en el formulario de ingreso de contribuyentes, se ingresaban letras en los campos numéricos, y se solucionó cambiando el tipo de dato a *integer* en la base de datos, también se ingresaban datos demasiado largos que arrojaban errores de acuerdo al tipo de datos, la solución fue cambiar el tipo de dato para el campo requerido.

Las pruebas fueron realizadas en un ordenador local con un procesador Intel Dual Core 2.3GHz y 3GB de memoria RAM.

Pruebas de Integración

Para realizar este tipo de prueba, fue necesario realizar una secuencia de procesos en los diferentes módulos.

A continuación, se detalla el proceso de ingreso de contribuyentes al sistema:

- Se ingresaron los datos en el formulario correspondiente dando como resultado un mensaje de éxito.
- Luego del ingreso, se le asignó un servicio de cuota fija al contribuyente.
- Se asigna una fecha de vencimiento a la zona correspondiente al contribuyente y se generan las boletas de cobro, al generarse las boletas se notó una incongruencia en el monto a pagar, solucionando este problema revisando el código en la clase PaymentCalculator y revisando la fórmula de cobro en el método calculatePaymentBasicBonus, la cual tenía una operación matemática errónea.

Pruebas del Sistema

Se realizaron las pruebas del sistema cuando se terminó el desarrollo de la aplicación web, ya que se incorporan los elementos de hardware, usuarios y la información que la aplicación manipulará, dichas pruebas se detallan a continuación:

- **Prueba de Seguridad.** Se realizó está prueba para determinar si la aplicación podría tener fallos y que algún usuario de ésta pueda acceder de forma malintencionada, por ejemplo, se accede al sistema y se intentó ingresar al módulo usuarios copiando y

pegando la URL del módulo a ingresar en una nueva ventana o navegador, y se pudo acceder a la pantalla de usuario de la cual se copia la dirección e incluso se puede eliminar un usuario.

La solución fue agregar en el código fuente al inicio de cada página que se hiciera la comprobación del tipo de usuario que ingresa a la aplicación, y si se detecta que dicha comprobación no es realizada, la aplicación carga la pantalla de inicio de sesión.

- **Prueba de Resistencia y Desempeño.** Se desarrolló está prueba para confrontar la aplicación con situaciones anormales como el hecho de escribir una lectura de contribuyente para servicio de tipo consumo con números mayores al máximo soportado en uso de espacio en el campo de la base de datos, arrojando un error, la solución fue cambiar el tipo de dato de las lecturas de *smallint* a *int*.

Pruebas de Aceptación

Estas iniciaron luego de la prueba de integración, terminando el ensamble de la aplicación web y después de descubrir y corregir los errores de interfaz, se centró en los procesos y salidas que el usuario final pudo obtener con la interacción. Se llevó a cabo el plan de capacitación.

Un plan de capacitación es una herramienta diseñada y estructurada con el propósito de guiar e informar al usuario final sobre el uso de una herramienta informática, en este caso enfocada al uso y manipulación la aplicación.

Dentro de este plan, se detalla la forma en que los nuevos usuarios serán instruidos en la utilización de las herramientas que están inmersas en la aplicación web, procurando con

ello, que cada usuario que tenga acceso a los módulos del sistema, pase de tener un trabajo manual, a uno automatizado.

4.6 PLAN DE IMPLEMENTACIÓN

En el plan de implementación se planifican las actividades necesarias para la instalación y puesta en marcha de la aplicación web, así como los recursos necesarios para su ejecución en la UATM de la Alcaldía Municipal de Chalchuapa.

4.6.1 Objetivos de la Implementación

General

Desarrollar una guía rápida de referencia para la instalación y configuración de las tecnologías utilizadas en el sistema SIGESAP.

Específicos

- Presentar los requerimientos que debe cumplir el servidor para la instalación del sistema y su base de datos.
- Mostrar los pasos para la instalación y configuración de los diferentes aplicativos.
- Planificar las capacitaciones a los usuarios para el posterior manejo del sistema.

4.6.2 Planeación de la Implementación.

Durante la implementación de la aplicación se establecerán una serie de actividades y procedimientos a realizar.

En dicho plan se seguirán procedimientos en los cuales intervengan el equipo desarrollador junto al personal de la UATM, de manera que interactúen con cada uno de los módulos de la aplicación web.

Las actividades que conlleva esta etapa incluyen la preparación del proyecto, instalación del sistema, presentación del proyecto, pruebas de aceptación y capacitación de la aplicación; para luego ponerlo en marcha.

En la figura, se presenta el plan de implementación del sistema y como está estructurada cada una de las actividades.

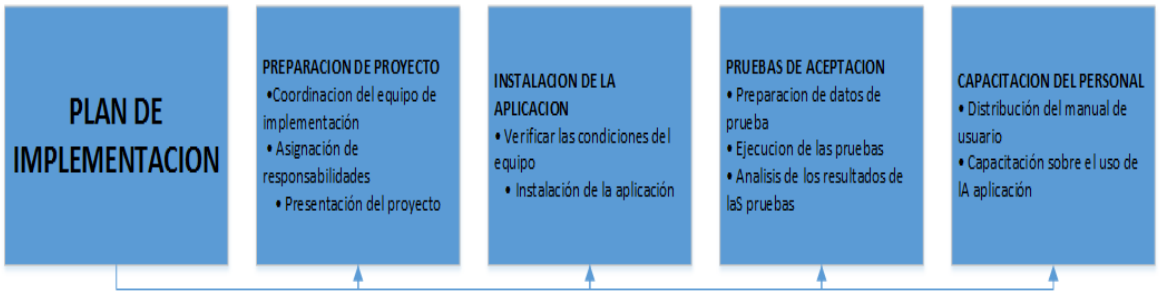


Figura 40: Diagrama del Plan de Implementación.

4.7 MANUALES DE USUARIO Y ADMINISTRADOR

Además del plan de implementación, el equipo de desarrollo entregará manuales de uso tanto para los usuarios del sistema como para el administrador. Estos manuales serán de mucha utilidad cuando se presenten dudas acerca del uso y configuración del sistema, también servirán de material de apoyo durante las capacitaciones.

4.7.1 Manual de Usuarios

Este manual contiene información que facilita el uso del sistema, el cual consta de:

- Una guía paso a paso de cómo realizar cada una de las tareas existente dentro del sistema informático.
- Una descripción de cada una de las funcionalidades e interfaces del sistema.

4.7.2 Manual de Administrador

Contiene información técnica y está dirigido a la persona encargada de instalar y dar mantenimiento al sistema

Este manual brinda soporte a las siguientes actividades:

- Muestra información referente a la instalación y configuración del sistema informático.
- Instalación de la base de datos.
- Configuración del servidor o estación de trabajo.
- Lista las instrucciones de programación usadas en el desarrollo del programa.
- Capacita al administrador para analizar, escribir y resolver errores.

CONCLUSIONES

- Es de vital importancia observar detalladamente los procesos actuales a la hora de diseñar un sistema. Una buena realización y un buen análisis del estudio previo es fundamental para obtener los requerimientos necesarios para el desarrollo de un proyecto informático, lo que minimiza los cambios en los requerimientos una vez se ha iniciado el proyecto.

- Con el pasar del tiempo, los procedimientos actuales de las instituciones comienzan a quedar desfasados, así que las instituciones deben evaluar periódicamente sus procesos para mantenerse en un funcionamiento óptimo.

- Para lograr alcanzar una fluidez en los procedimientos de las organizaciones, se debe prestar especial atención a la creación de documentos que estandaricen los respectivos procesos de la institución.

- Además del diseño de los elementos que intervienen directamente con la solución del problema, también es de suma importancia tomar en cuenta otros componentes como lo son la seguridad del sistema y un buen diseño de la interfaz de usuario.

- La aplicación reducirá significativamente el tiempo involucrado en cada actividad del proceso actual de administración y gestión de servicios de agua de la UATM. Se reducirá notablemente el tiempo y esfuerzo invertido en la obtención y consolidación de la información, ya que el sistema se encargará de mostrar dicha información de manera más fácil y rápida.

RECOMENDACIONES

- Para la puesta en marcha del sistema, se recomienda tomar como guía el manual de administrador técnico, de forma que facilite la configuración del entorno para que la implementación del sistema sea satisfactoria.

- Se sugiere a la unidad de informática que por seguridad se encargarse de la creación periódica de respaldos (backups) de la base de datos.

- A los usuarios del sistema, que deben de hacer uso de los manuales contenidos en el CD del proyecto para tener una mejor comprensión del funcionamiento del sistema informático.

- Estandarizar los procesos de la unidad y documentarlos en manuales de procedimientos para el fácil análisis de estos. De la misma forma evaluar los procedimientos actuales para detectar deficiencias e implementar mejoras en los procesos.

- Dar mantenimiento al equipo de cómputo tanto en el área de informática como en la Unidad de Administración Tributaria Municipal para que este en óptimas condiciones para el funcionamiento del sistema.

BIBLIOGRAFÍA

- Kendall, K., & Kendall, J. (2011). Análisis y Diseño de Sistemas. México: Pearson Educación.

- Tapia, Nestor. (2019). Ventajas y desventajas del lenguaje PHP. Obtenido dehttps://www.baulphp.com/ventajas-y-desventajas-del-lenguaje-php/

- Hypertextual. (Mayo, 2013).Entendiendo HTML5: guía para principiantes. Obtenido de https://hipertextual.com/archivo/2013/05/entendiendo-html5-guia-para-principiantes

- CulturaChalchuapaneca. (Octubre, 2019). Chalchuapa un lugar para visitar. Obtenido de http://culturachalchuapaneca.blogspot.com/2009/10/historia-de-chalchuapa.html

- Baquero García, Jose M. (Diciembre, 2015). ¿Qué es Laravel? Obtenido de https://www.arsys.es/blog/programacion/que-es-laravel

- TutorialsPoint. (s.f.). Software - Ciclo de Vida de Desarrollo. Obtenido de https://www.tutorialspoint.com/es/software_engineering/software_development_life_cycle.htm

- Laragon. (Marzo, 2019). WhyLaragon? Obtenido de https://laragon.org/why-laragon

- Cabana, Judit. (Agosto, 2017). Editores de código: Sublime Text y PHPStorm. Obtenido de https://www.drauta.com/editores-de-codigo-sublime-text-y-phpstorm

- Alvarez, Miguel Angel. (Abril, 2018). Autenticación por token. Obtenido de https://desarrolloweb.com/articulos/autenticacion-token.html

- Arevalo Lizardo, Maria Eugenia. (Noviembre, 2012). Propuesta de Estándar de desarrollo o codificación (Primera Entrega). Obtenido de https://arevalomaria.wordpress.com/2012/11/02/propuesta-de-estandar-de-desarrollo-o-codificacion-primera-entrega-programacion

- Strong, E. D. (Julio, 2019). How Many Watts Does a Computer Use? Obtenido de https://www.techwalla.com/articles/how-many-watts-does-a-computer-use

- AES El Salvador. (Julio, 2019). Tarifas vigentes del servicio eléctrico. Obtenido de http://www.aes-elsalvador.com/site/assets/files/1168/tarifas-clesa.jpg

GLOSARIO

A

Abstracción: Consiste en aislar un elemento de su contexto o del resto de los elementos que lo acompañan.

API: Interfaz de Programación de Aplicacioneses un conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software.

B

Backup: Es una copia de los datos originales que se realiza con el fin de disponer de un medio para recuperarlos en caso de su pérdida.

Biblioteca: Es un conjunto de implementaciones funcionales, codificadas en un lenguaje de programación, que ofrece una interfaz bien definida para la funcionalidad que se invoca.

C

CRUD: Es el acrónimo de "Crear, Leer, Actualizar y Borrar", que se usa para referirse a las funciones básicas en bases de datos o la capa de persistencia en un software.

D

DBA: Administrador de Bases de Datos es aquel profesional que administra las tecnologías de la información y la comunicación, siendo responsable de los aspectos técnicos,

tecnológicos, científicos, inteligencia de negocios y legales de bases de datos, y de la calidad de datos.

Debugging: Es el proceso de identificar y corregir errores de programación.

Diagrama Entidad Relación: Es una herramienta para el modelado de datos que permite representar las entidades relevantes de un sistema de información, así como sus interrelaciones y propiedades.

E

Encapsulamiento: Se denomina al ocultamiento del estado, es decir, de los datos miembros de un objeto de manera que solo se pueda cambiar mediante las operaciones definidas para ese objeto.

Encriptar: Técnicas de cifrado o codificado destinadas a alterar las representaciones lingüísticas de ciertos mensajes con el fin de hacerlos ininteligibles a receptores no autorizados

F

Framework: Es un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

Front-End: En diseño web (o desarrollo web) hace referencia a la visualización del usuario navegante, se refiere a la parte del sistema que convierte la entrada del texto en una representación simbólica.

H

Herencia: Es el mecanismo más utilizado para alcanzar algunos de los objetivos más preciados en el desarrollo de software como lo son la reutilización y la extensibilidad.

I

Indexación: Se refiere a diversos métodos para incluir en el índice de internet el contenido de un sitio web.

Instancia: Es la particularización, realización específica u ocurrencia de una determinada clase, entidad o prototipo.

M

Metadatos: Se refiere a un grupo de datos que describen el contenido informativo de un objeto.

O

Ordenanza Municipal: Es un tipo de norma jurídica que se incluye dentro de los reglamentos, y que son dictadas por un ayuntamiento o municipalidad para la gestión de su municipio.

P

Paradigma: Conjunto de creencias, prácticas y conocimientos que guían el desarrollo de una disciplina durante un período de tiempo.

Parsing: Es un programa informático que analiza una cadena de símbolos de acuerdo a las reglas de una gramática formal.

POO: Programación Orientada a Objetoses un paradigma de programación que viene a innovar la forma de obtener resultados. Los objetos manipulan los datos de entrada para la obtención de datos de salida específicos, donde cada objeto ofrece una funcionalidad especial.

Pseudocódigo: Es una descripción de alto nivel compacta e informaldel principio operativo de un programa informático u otro algoritmo.

R

Refactorizar: Es una técnica de la ingeniería de software para reestructurar un código fuente, alterando su estructura interna sin cambiar su comportamiento externo.

S

Software Libre: Es todo programa informático cuyo código fuente puede ser estudiado, modificado, y utilizado libremente con cualquier fin y redistribuido sin o con cambios y/o mejoras.

SRS: Especificación de Requisitos de Software es una descripción completa del comportamiento del sistema que se va a desarrollar.

U

UML: Lenguaje Unificado de Modelado es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema.

UPS: Sistema de Alimentación Ininterrumpida es un dispositivo que gracias a sus baterías u otros elementos almacenadores de energía, durante un apagón eléctrico puede proporcionar energía eléctrica por un tiempo limitado a todos los dispositivos que tenga conectados.

URL: Localizador de Recursos Uniforme es una cadena de caracteres con la que se asigna una dirección única a cada uno de los recursos de información disponibles en Internet.

ANEXOS

Anexo 1: Ordenanza Reguladora de Tasas por Servicios Municipales (Fragmento)

No. 10	**SERVICIO DE AGUA POTABLE**	
A.	Por conexión de cada servicio en la red municipal	
A.1	Cuando no requiera rompimiento de calle	$ 60.00
A.2	Cuando requiera rompimiento de calle	$ 85.00
B.	Consumo del servicio de agua	
B.1	Cuota básica mensual por consumo de hasta 20 metros cúbicos,	
		6
	registrado en medidor (1)	$ 4.00
B.2	Metro cúbico registrado en medidor, en exceso de 20 metros cúbicos mensuales (1)	$ 0.25
B.3	Metro cúbico registrado en medidor, para uso agroindustrial	$ 0.25
B.4	En ninguno de los tres casos anteriores la tasa mensual por el uso de este servicio podrá ser inferior a	$ 1.00
B.5	Cuota fija mensual para usuarios sin medidor	$ 4.00
B.6	Derecho mensual por poseer mecha del servicio de agua potable (1)	$ 1.00
C.	Por desconexión del servicio de agua a solicitud del interesado	$ 12.00
D.	Por reconexión del servicio de agua a solicitud del interesado	$ 12.00

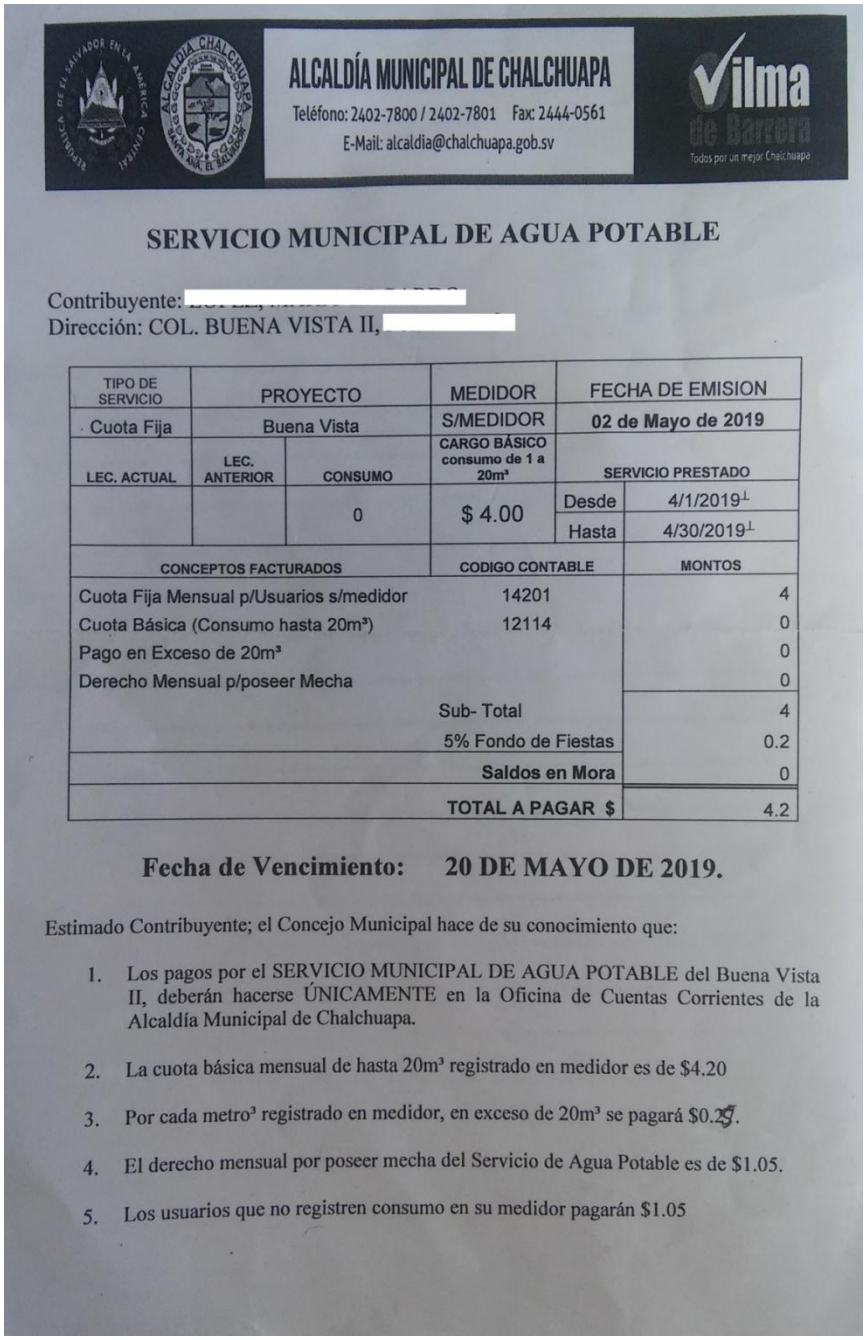
Cuando el contribuyente del servicio de agua potable, haya dejado de pagar por dos meses consecutivos, tendrá que pagar multa e intereses por mora; y después de cuatro meses consecutivos sin haber hecho efectivo el pago, se le suspenderá el servicio, y su reconexión se realizará después de pagada la mora y sus accesorios, más el correspondiente derecho por reconexión.

Una vez suspendido el servicio, el usuario que ilegalmente realice la reconexión pagará una multa de $ 150.00 y se le suspenderá el servicio; el cual será reconectado cuando haya cancelado la multa, la mora más sus accesorios si existieren; así como el respectivo derecho de reconexión establecido en el literal "D" de este numeral.

Cuando ilegalmente una persona efectúe la conexión del servicio, pagará una multa de $ 150.00 y el servicio será suspendido; el cual podrá conectarse legalmente luego de haber pagado la multa más el respectivo derecho de conexión según corresponda de acuerdo al literal "A" del presente numeral.

Las reparaciones, conexiones, reconexiones o suspensiones del servicio de agua que se efectúen sobre la red municipal; serán realizadas única y exclusivamente por personal de la municipalidad debidamente autorizado. Cualquier persona natural o jurídica que contravenga esta disposición, se hará acreedor de las multas correspondientes; sin perjuicio de la multa por el rompimiento de calle sin autorización y de las acciones penales correspondientes.

Anexo 2: Boleta de cobro por servicios de agua potable (Actual)



ALCALDÍA MUNICIPAL DE CHALCHUAPA
Teléfono: 2402-7800 / 2402-7801 Fax: 2444-0561
E-Mail: alcaldia@chalchuapa.gob.sv

Vilma de Barrera
Todos por un mejor Chalchuapa

SERVICIO MUNICIPAL DE AGUA POTABLE

Contribuyente: [redacted]
Dirección: COL. BUENA VISTA II, [redacted]

TIPO DE SERVICIO	PROYECTO		MEDIDOR	FECHA DE EMISION	
Cuota Fija	Buena Vista		S/MEDIDOR	02 de Mayo de 2019	
LEC. ACTUAL	LEC. ANTERIOR	CONSUMO	CARGO BÁSICO consumo de 1 a 20m³	SERVICIO PRESTADO	
		0	$ 4.00	Desde	4/1/2019
				Hasta	4/30/2019

CONCEPTOS FACTURADOS	CODIGO CONTABLE	MONTOS
Cuota Fija Mensual p/Usuarios s/medidor	14201	4
Cuota Básica (Consumo hasta 20m³)	12114	0
Pago en Exceso de 20m³		0
Derecho Mensual p/poseer Mecha		0
	Sub- Total	4
	5% Fondo de Fiestas	0.2
	Saldos en Mora	0
	TOTAL A PAGAR $	4.2

Fecha de Vencimiento: 20 DE MAYO DE 2019.

Estimado Contribuyente; el Concejo Municipal hace de su conocimiento que:

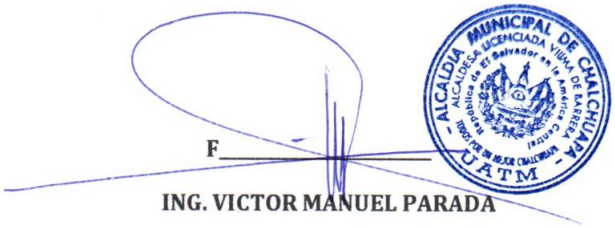
1. Los pagos por el SERVICIO MUNICIPAL DE AGUA POTABLE del Buena Vista II, deberán hacerse ÚNICAMENTE en la Oficina de Cuentas Corrientes de la Alcaldía Municipal de Chalchuapa.
2. La cuota básica mensual de hasta 20m³ registrado en medidor es de $4.20
3. Por cada metro³ registrado en medidor, en exceso de 20m³ se pagará $0.25.
4. El derecho mensual por poseer mecha del Servicio de Agua Potable es de $1.05.
5. Los usuarios que no registren consumo en su medidor pagarán $1.05

Anexo 3: Carta de conformidad.

Chalchuapa, Santa Ana ,25 de julio de 2019

EL SUSCRITO JEFE DE LA UNIDAD DE LA ADMINISTRACION TRIBUTARIA MUNICIPAL. En uso de sus facultades legales, y vista la solicitud presentada por los jóvenes: VÍCTOR ISMAEL MARTÍNEZ RODRÍGUEZ, ERICK ADIEL TRIGUEROS JERÉZ, GUSTAVO ADOLFO VELÁSQUEZ PLEITEZ, estudiantes egresados de la Carrera de Ingeniería de Sistemas Informáticos de la Universidad de El Salvador Facultad Multidisciplinaria de Occidente. HACE CONSTAR: Que HAN REALIZADO UNA EXCELENTE LABOR EN BENEFICIO DE LA MUNICIPALIDAD, Por medio de la creación y presentación de un Nuevo Sistema Informático denominado SIGESAP para el cobro del servicio de AGUA POTABLE de las diferentes comunidades.

Y Para los usos que estimen conveniente los interesados, se extiende la presente, en la Alcaldía Municipal de Chalchuapa, Departamento de Santa Ana, a los veinticinco días del mes de Julio del dos mil diecinueve.



F_____

ING. VICTOR MANUEL PARADA

JEFE DE UATM.-