PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

FACULTAD DE INGENIERÍA

SISTEMAS DE INFORMACIÓN

TRABAJO DE TITULACIÓN

DESARROLLO DE UNA APLICACIÓN WEB PARA LA ADMINISTRACIÓN DE COBRO A ARRENDATARIOS. CASO DE ESTUDIO: URBANIZACIÓN LA LUZ

AUTOR:

KEVIN ALEXANDER ORMAZA GÁLVEZ

QUITO, 2023

DEDICATORIA

Este trabajo está dedicado a mi familia, quienes han sido testigos y participantes de mi viaje académico. Cada página es un tributo a su sacrificio y dedicación, un recordatorio eterno de que mi éxito es su éxito.

En especial a mi amada madre y hermana, cuyo inquebrantable apoyo y aliento han sido mi faro en los momentos oscuros. Agradezco sus sabias palabras y la guía constante que han marcado mi camino a lo largo de esta travesía académica.

Dedico este trabajo a mi querida abuelita Margarita, cuya sabiduría y amor han sido el cimiento de mi existencia. Y a mis tíos y tías, quienes, con su constante vigilancia, han sido los pilares que me han impedido caer.

AGRADECIMIENTO

Agradezco principalmente a mi querida madre y hermana, cuyo apoyo incondicional ha sido mi mayor fortaleza a lo largo de esta travesía académica. Cada logro en este trabajo es un testimonio de su amor y compromiso hacia mi educación, y no tengo palabras suficientes para expresar la gratitud que siento por tenerlas a mi lado.

No quiero pasar por alto el invaluable apoyo de mis amigos, quienes han estado a mi lado en cada proyecto y aventura. Su constante aliento y respaldo han sido un regalo preciado, y agradezco sinceramente su amistad y apoyo incondicional.

Agradezco sinceramente a mis profesores y tutores por su dedicación y enseñanzas, que han sido fundamentales en mi formación académica. Este trabajo es el fruto de su influencia, y reconozco con gratitud el impacto positivo que han tenido en mi desarrollo como estudiante y como persona.

Finalmente, dedico un agradecimiento especial a todas las personas mencionadas y no mencionadas que, de una forma u otra, han contribuido al éxito de este proyecto. Este logro es el resultado de un esfuerzo colectivo.

RESUMEN

En el marco de la gestión de cobros de cuotas de departamentos en Ecuador, se ha desarrollado un aplicativo web integral para optimizar este proceso crucial. El presente trabajo de titulación presenta una aplicación web desarrollada con el marco Laravel y basada en la arquitectura Modelo-Vista-Controlador (MVC).

La metodología empleada se sustenta en un enfoque estructurado que parte de la planificación y diseño detallado. La implementación técnica se lleva a cabo en fases, desde la configuración del proyecto en Laravel hasta la creación de interfaces intuitivas y sistemas de autenticación robustos. Las pruebas de aceptación desempeñan un papel esencial en garantizar la calidad del software, validando cada funcionalidad, desde el registro de usuarios hasta la generación de informes mensuales y anuales.

El trabajo destaca la importancia de la tecnología en la optimización de procesos administrativos, específicamente en el ámbito de la gestión financiera de urbanizaciones. Se ofrecen recomendaciones para el mantenimiento continuo de la aplicación, la atención eficiente a usuarios y la consideración de la escalabilidad del aplicativo para adaptarse a posibles aumentos en el número de usuarios o datos. Este aplicativo no solo aborda las necesidades actuales de la gestión de cobros, sino que sienta las bases para un enfoque tecnológico sostenible y adaptable a futuras demandas.

ÍNDICE

Tabla de contenido

ÍNDICE DE FIGURAS, GRÁFICOS Y TABLAS

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

SINOPSIS DE LOS CAPÍTULOS

El primer capítulo del proyecto destaca la urgente necesidad de mejorar la gestión de cuotas en las urbanizaciones de Ecuador, señalando desafíos como la falta de transparencia y la ineficiencia en los procesos. Se plantean preguntas de investigación para abordar estos problemas, junto con objetivos específicos que delinean el alcance del proyecto.

El segundo capítulo presenta los conceptos fundamentales que respaldan el desarrollo del proyecto en la Urbanización La Luz. Se exploran aspectos teóricos y conceptuales clave, esenciales para comprender la problemática de la comunidad y la solución propuesta. Cada sección se centra en un elemento clave que influye en el diseño e implementación de la aplicación web destinada a mejorar la gestión financiera de la urbanización.

En el Capítulo III, se aborda en detalle la creación y construcción de la aplicación web diseñada para la administración de cobros de cuotas en urbanizaciones. Este capítulo se estructura en diversas fases, cada una con un propósito específico en el ciclo de desarrollo de software.

El capítulo IV finaliza el proyecto de la aplicación web para el control de cobro de cuotas en departamentos, dirigida a propietarios y arrendatarios en urbanizaciones. Las conclusiones destacan el impacto positivo de la aplicación al mejorar la transparencia y eficiencia en la gestión de recursos financieros. La capacidad del sistema para realizar un seguimiento detallado de pagos y gastos facilita la toma de decisiones informadas.

1. CAPÍTULO I: INTRODUCCIÓN

En este primer capítulo, se presenta un contexto crucial que sirve como base para el proyecto en curso. Se enfatiza la urgente necesidad de mejorar la gestión de las cuotas en las urbanizaciones de Ecuador, ya que estas cuotas desempeñan un papel fundamental en el mantenimiento de las áreas comunes, los servicios y los proyectos de infraestructura. No obstante, se identifican desafíos significativos, incluyendo la falta de transparencia, la ineficiencia en los procesos y la ausencia de información detallada. Estos desafíos se traducen en una serie de preguntas de investigación destinadas a abordar este problema. Asimismo, se establecen los objetivos del proyecto y se define su alcance, sentando las bases para la implementación de una solución tecnológica que tenga como objetivo mejorar la calidad de vida de los residentes y optimizar la gestión financiera en las urbanizaciones.

1.1. Justificación

En Ecuador el cobro de cuotas de departamentos es una práctica común, estas cuotas sirven para financiar el mantenimiento de las áreas comunes, servicios de limpieza, servicios básicos, seguridad o costear proyectos de mejora de la infraestructura como instalación de cámaras o renovar instalaciones eléctricas. Estos cobros se realizan de manera mensual, trimestral o según la política de la urbanización. Es importante desarrollar un aplicativo que automatice este proceso ya que se busca solucionar varios problemas como la falta de transparencia al momento de los pagos, gestión ineficiente al momento de la ejecución de proyectos o falta de información clara y detallada sobre los plazos.

La modernidad demanda automatizar los servicios existentes para asegurar la comodidad de los residentes en las urbanizaciones. Esto se logra a través de la implementación de tecnología que mejora la eficiencia en la administración del cobro de cuotas y mantenimiento de los departamentos.

1.2. Planteamiento del problema.

En las urbanizaciones existe un desafío al momento del cobro de las cuotas ya que depende de varias personas y procesos. Además hay problemas que exigen una solución tecnológica como la comunicación entre administradores y propietarios o arrendatarios de los departamentos, falta de compromiso lo que llevaría a problemas de pago y por lo tanto

afección a la planificación de los gastos de la urbanización, falta de información como saber quién es el encargado del cobro, esto llevaría a retrasos en los cobros.

En función de estas problemáticas se plantea las siguientes preguntas principales:

- ¿Cómo se pueden mejorar los procesos de gestión y cobro de cuotas para el mantenimiento de las urbanizaciones?
- ¿Cómo se podría llevar la administración de cada departamento detallada?

Y las siguientes preguntas secundarias:

- ¿Cuáles son los principales problemas que enfrentan los administradores de urbanizaciones en el proceso de cobro de cuotas y mantenimiento?
- ¿Cuál es el nivel de satisfacción de los propietarios de departamentos en cuanto a los procesos de cobro de cuotas y mantenimiento en su urbanización?
- ¿Qué herramientas tecnológicas están disponibles actualmente para mejorar los procesos de gestión y cobro de cuotas de mantenimiento en urbanizaciones?

1.3. Objetivos.

1.3.1. Objetivo General.

Desarrollar una aplicación web, que permita la administración eficiente de cobros de cuotas en urbanizaciones, utilizando el framework Laravel en conjunto con herramientas como MySQL, XAMPP y Visual Studio Code. El desarrollo incluirá funcionalidades específicas, como la gestión de usuarios, el registro de pagos y la generación de reportes.

1.3.2. Objetivos Específicos.

- Identificar requerimientos
- Realizar el diseño del aplicativo a desarrollar
- Aplicar una metodología de Desarrollo XP.
- Diseñar interfaces intuitivas y fáciles de usar para el usuario final.
- Realizar reportes que consten ingresos y egresos.

1.4. Alcance.

Este proyecto se enfocará en desarrollar una versión inicial del aplicativo web de gestión para propietarios y arrendatarios de departamentos en urbanizaciones. El desarrollo incluirá funcionalidades clave, como la gestión de usuarios, el registro de pagos y la generación de reportes. La versión inicial del aplicativo proporcionará una base sólida para

mejorar la transparencia y eficiencia en la gestión de recursos financieros en las urbanizaciones y sentará las bases para futuras expansiones y mejoras.

2. CAPÍTULO II: MARCO TEÓRICO

En este capítulo se proporcionarán los conceptos principales que sustentan el desarrollo del proyecto en la Urbanización La Luz. A lo largo de este capítulo, se explorará una serie de aspectos teóricos y conceptuales fundamentales que son esenciales para comprender la problemática que enfrenta la comunidad y la solución propuesta. Cada sección se enfoca en un elemento clave que influye en el diseño y la implementación de la aplicación web destinada a mejorar la gestión financiera de la urbanización.

2.1. Marco Espacial

El marco espacial del proyecto es la Urbanización La Luz, ubicada en la ciudad de Quito, en el barrio La Luz, sector norte de la ciudad. Su considerable cantidad de departamentos y bloques residenciales la convierte en el objeto central de estudio. La cercanía de esta urbanización a mi lugar de residencia me proporciona una comprensión más profunda de los problemas y desafíos que enfrenta nuestra comunidad.

Para este proyecto, realicé una investigación exhaustiva del sistema actual de cobro de cuotas en la Urbanización La Luz, con el objetivo de mejorar la gestión financiera de la comunidad y abordar la falta de transparencia y eficiencia en la administración de los recursos financieros. Esta investigación me motivó a desarrollar una aplicación web que automatice este proceso.

La Urbanización se enfrenta a desafíos relacionados con la comunicación y la claridad en la asignación de los fondos recaudados. Con este proyecto, aspiro a brindar una solución que incremente la transparencia y mejore la calidad de vida de los residentes, asegurando un uso más eficiente de los recursos financieros de la Urbanización. Este enfoque en la mejora de la administración financiera se presenta como esencial para mantener y elevar la calidad de vida en nuestra comunidad.

2.2. Administración de Cobros

La administración de cobros en el sector inmobiliario es un proceso que implica la recolección de pagos en efectivo o transferencia y la gestión financiera de las urbanizaciones.

La gestión de deudas es un proceso crucial en la administración de cobros. Cuando los arrendatarios o propietarios no cumplen con sus pagos se deben implementar acciones adecuadas para recuperar los fondos pendientes, así lo resume (Oviedo, 2013): "Las cuotas de urbanización son un ingreso afectado que, sirve para financiar la urbanización de determinadas áreas, sin embargo, su regulación es escasa y difusa. De hecho en primer lugar hay que acudir a la legislación urbanística y de forma supletoria a la legislación tributaria. Encontrándonos en determinados momentos con lagunas sobre la forma de emisión, los plazos de cobro e incluso los conceptos por los que se puede pasar la cuota. En este trabajo se pretenden disipar las lagunas que en la práctica surgen cuando se emiten cuotas de urbanización en el sistema de cooperación."

La gestión eficiente implica establecer procesos sólidos para registrar y dar seguimiento a los pagos de los arrendatarios o propietarios, asegurando que las cuotas se paguen puntualmente.

2.3. Metodología de desarrollo XP

La metodología de desarrollo XP (Extreme Programming) es una metodología ágil de desarrollo de software que se basa en la comunicación y realimentación del código desarrollado y su objetivo según (Fernández Escribano, 2009) es: "aumentar la productividad al desarrollar software con una fuerte arquitectura, intentando sacar productos al mercado rápidamente, con gran calidad y motivando al equipo de trabajo para seguir mejorando esta tendencia." Las bases en las que esta metodología se fundamenta según (Fernández Escribano, 2009) son:

- ✓ **Disponibilidad del usuario:** Se espera que los clientes estén disponibles para la interacción con el equipo de desarrollo constantemente para una comunicación fluida y comprensión profunda de los requerimientos del usuario.
- ✓ **Estándares de implementación:** Se refiere a la importancia de seguir estándares de codificación coherentes para garantizar uniformidad en la estructura del código.
- ✓ **Unidades de prueba o test:** Esta metodología promueve la escritura de pruebas automatizadas antes de escribir el código de producción. Esto con la finalidad de que el código funcione según lo previsto y permite realizar pruebas continuas para detectar y corregir problemas de manera temprana.
- ✓ **Integración del código:** Se promueve la integración continua del código al repositorio compartido. Esto asegura que los cambios realizados por diferentes

miembros del equipo se fusionen y se prueben de manera regular, lo que reduce el riesgo de conflictos y errores.

- ✓ **Frecuencia en la integración del código:** La metodología XP enfatiza la integración frecuente del código, idealmente varias veces al día. Esto facilita la detección temprana de problemas y la entrega regular de versiones funcionales del software.
- ✓ **El código es propiedad de todos:** En XP, el código no es propiedad exclusiva de un desarrollador individual. Todos los miembros del equipo tienen la responsabilidad de revisar, mantener y mejorar el código. Esto fomenta la colaboración y la responsabilidad compartida.
- ✓ **Dejar las optimizaciones para el final:** En lugar de buscar la optimización extrema desde el principio, XP sugiere centrarse en la funcionalidad y la calidad del código. Las optimizaciones se abordan en una etapa posterior para no comprometer la productividad temprana.
- ✓ **No a las horas extras:** XP valora la eficiencia y el equilibrio entre el trabajo y la vida personal. Se desalienta el trabajo excesivo o las horas extras forzadas, ya que esto puede llevar a una disminución de la calidad del trabajo y el agotamiento del equipo.

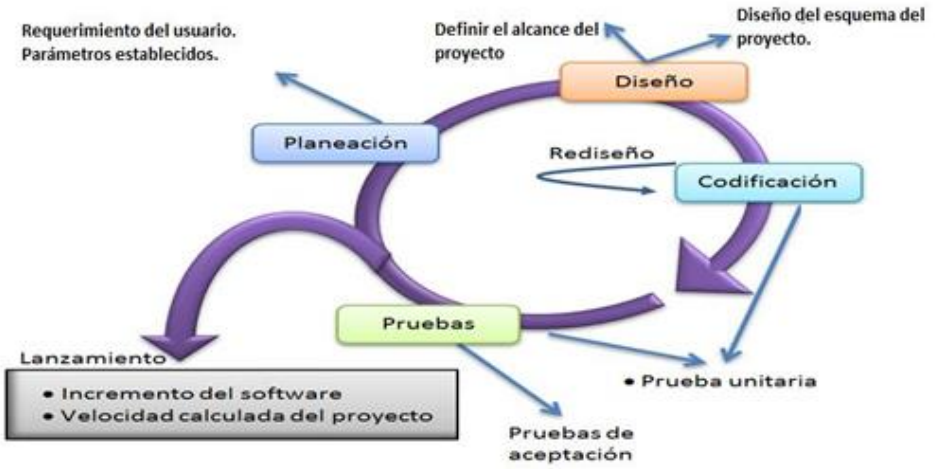

Ilustración 1: Ciclo de vida de la metodología Extreme Programming, según (Rodriguez, 2018).

Según la (Ilustración 1), el ciclo de vida de la metodología XP consta de varias fases o series que se aplican de manera continua a lo largo del proyecto, en las que se encuentran:

- Fase de Planificación: Según (Pérez, 2011): “Los clientes plantean a grandes rasgos las historias del usuario que son de interés para la primera entrega del producto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del aplicativo, construyendo un prototipo.” En esta fase se realiza el diseño de alto nivel y se comienza a definir la arquitectura del sistema, además las ideas y las sugerencias pospuestas se documentan para una puesta en práctica posterior.
- Fase de Diseño: En esta fase se seleccionan las historias de usuario para la iteración inicial y se estiman los tiempos y recursos necesarios.
- Fase de Codificación: En esta fase los programadores implementan las historias de usuario seleccionadas en la iteración actual. Se enfatiza la programación en parejas, donde dos desarrolladores trabajan juntos en el mismo código.
- Fase de Pruebas: La fase de pruebas constituye un paso crítico en el proceso de desarrollo de software. Durante esta etapa, se realizan pruebas exhaustivas para garantizar que el sistema funcione correctamente y cumpla con los requisitos definidos en las fases anteriores. Entre las pruebas del aplicativo más importantes se encuentran:
    - Pruebas de aceptación: Son pruebas realizadas por el cliente o los usuarios finales para verificar que el software cumple con sus requisitos. Se realizan al final del proceso de desarrollo de software, antes de que el software se lance a producción.
    - Pruebas Unitarias: Son pruebas que se realizan para verificar que una unidad de código funciona correctamente. Una unidad de código puede ser una función, un método, una clase o un componente.

El proceso concluye con el lanzamiento. En esta etapa, el software se implementa en el entorno de producción o se pone a disposición de los usuarios finales. Esto puede implicar la instalación en servidores, la configuración de bases de datos y la capacitación del personal. El lanzamiento también puede incluir la promoción del software y la comunicación con los usuarios para informarles sobre las nuevas características o mejoras.

2.4. Desarrollo Web

2.4.1. Back-end

El desarrollo back-end se encarga de la parte lógica de una página web para que sea funcional, además se centra en todos los aspectos técnicos de los productos y se enfoca en

procesar información o datos para que el front-end trabaje con éxito. Además, la persona que se encarga de este desarrollo se denomina “desarrollador back-end”, esta persona es quién: “diseña, construye y mejora el servicio de la información administrada como base de datos, creando una experiencia de navegación ágil, dinámica e interactiva para el usuario, en conjunción a los códigos gestionados por el frontend.” (Quintana, 2023).

El back-end actúa como la columna vertebral que sostiene la funcionalidad, la seguridad y la eficiencia del aplicativo. El backend administra los datos, la lógica de negocio y la interacción con las bases de datos, lo que garantiza la integridad de la información, la escalabilidad para manejar un mayor flujo de usuarios y datos, y la implementación de medidas de seguridad esenciales para proteger los datos sensibles.

El desarrollador back-end desempeña un papel fundamental en la construcción de una página web robusta y funcional. Además de diseñar, construir y optimizar los servicios de información administrados, este profesional se esfuerza por crear una experiencia de navegación ágil, dinámica y altamente interactiva para los usuarios. Trabaja en estrecha colaboración con los desarrolladores front-end para garantizar que la parte visible del sitio web funcione de manera armoniosa con la lógica y los datos gestionados en el back-end. Esta colaboración se traduce en una experiencia de usuario fluida y satisfactoria. Por lo tanto, el desarrollador back-end desempeña un papel esencial como el arquitecto que da forma a la infraestructura detrás del sitio web, asegurando que todos los componentes funcionen en conjunto de manera efectiva y eficiente.

2.4.2. Front-end

El front-end es el sistema desde el punto de vista del usuario, esto se refiere a todo lo que conlleva una página web, es decir, barras de búsqueda, botones, logotipo, formularios, widgets, etc. Además (Mestres, 2015), lo define como: “conjunto de tecnologías que se encargan de la presentación e interacción de una aplicación web con el usuario. El front-end se compone principalmente de tres lenguajes: HTML, CSS y JavaScript, que se ejecutan en el navegador web del usuario y permiten crear páginas web con estructura, estilo y funcionalidad.”

Al momento de desarrollar una interfaz frontal de calidad es imprescindible asegurar una experiencia positiva al usuario para que alcance el éxito digital. Existen aspectos muy importantes como la apariencia atractiva, usabilidad y adaptabilidad que significan una

retención de usuarios, vistas de los clientes y visibilidad de la marca. Un entorno digital altamente competitivo, el desarrollo exitoso de un front-end emerge como un factor distintivo fundamental y aporta a la prosperidad y al profesionalismo de una página web o aplicación.

El desarrollo front-end, como parte esencial del proceso de creación de una página web, se enfoca en la experiencia del usuario y en la presentación de la aplicación web. Esto abarca la creación de elementos visuales como barras de búsqueda, botones, logotipos, formularios y widgets que los usuarios interactúan directamente. Con un enfoque en la presentación y la interacción, el front-end se encarga de la apariencia atractiva y la usabilidad de la página web, asegurando una experiencia positiva para el usuario. La adaptabilidad a diferentes dispositivos y pantallas también es un componente crucial para garantizar que la página web sea accesible y atractiva en múltiples contextos. En un entorno digital altamente competitivo, la calidad del front-end emerge como un factor distintivo esencial que contribuye al éxito y la visibilidad de una marca en línea, así como a la satisfacción y retención de los usuarios.

2.5. Lenguaje de programación PHP

Según (Hills & Klint, 2017) PHP es: "un lenguaje de programación compatible, escalable, seguro y multidisciplinario que permite el desarrollo de aplicaciones ágiles, óptimas e inmediatas en función de los requerimientos de la sociedad. Hoy en día, PHP ha emergido como uno de los lenguajes de programación más destacados y de uso extendido dentro de la comunidad de código abierto, principalmente en el ámbito de la creación de aplicaciones web de considerable envergadura.

Una de las ventajas de PHP es su facilidad de uso. Además de su facilidad de aprendizaje y su sintaxis sencilla y similar a la de otros lenguajes. Otro aspecto es su gran cantidad de frameworks y bibliotecas, y una comunidad activa de desarrolladores que proporciona recursos y soluciones en línea. Su capacidad para interactuar con servidores web populares y su historia de madurez lo convierten en una herramienta confiable y adaptable para una amplia gama de proyectos web, desde sitios simples hasta aplicaciones empresariales complejas.

El lenguaje de programación PHP se destaca como un recurso versátil y esencial en el desarrollo de aplicaciones web. PHP es reconocido por su compatibilidad, escalabilidad y

seguridad, lo que lo convierte en una elección óptima para abordar una amplia gama de requerimientos de la sociedad. Su adopción en el ámbito de código abierto ha sido notable, especialmente en el desarrollo de aplicaciones web de gran envergadura. Una de las características más notables de PHP es su facilidad de uso, con una sintaxis amigable y familiar para quienes tienen experiencia en otros lenguajes de programación. Además, PHP cuenta con una amplia gama de frameworks y bibliotecas que aceleran el proceso de desarrollo y una comunidad activa de desarrolladores que ofrece recursos y soluciones en línea. Su capacidad para interactuar eficazmente con servidores web populares y su historial de estabilidad hacen de PHP una herramienta fiable y altamente adaptable, adecuada para proyectos que van desde sitios web simples hasta aplicaciones empresariales complejas.

2.6. Framework de desarrollo web (Laravel)

Laravel es un framework de PHP, el cual permite el desarrollo de aplicaciones web personalizadas y de alta calidad. Es conocido con por su elegante sintaxis y su énfasis en la simplicidad. Entre sus principales ventajas se encuentran:

- ✓ **Sintaxis clara:** Laravel se destaca por su sintaxis limpia y legible, lo que facilita la escritura de código claro y comprensible. Esto ayuda a los desarrolladores a ser más productivos y a reducir errores.
- ✓ **Migraciones de base de datos:** Laravel proporciona un sistema de migraciones de base de datos que permite a los desarrolladores definir y modificar la estructura de la base de datos de manera programática.
- ✓ **Seguridad:** Laravel se preocupa por la seguridad y ofrece características integradas para proteger las aplicaciones web. Incluye protección contra ataques CSRF (Cross-Site Request Forgery), autenticación integrada y manejo de contraseñas de manera segura.
- ✓ **Comunidad Activa:** Laravel cuenta con una comunidad activa y comprometida de desarrolladores que proporcionan soporte, recursos y extensiones. Esto hace que sea más fácil encontrar soluciones a problemas comunes y mantenerse al día con las mejores prácticas.
- ✓ **Sistema de enrutamiento:** El sistema de enrutamiento de Laravel es flexible y poderoso. Permite definir rutas de manera clara y mapearlas a controladores, lo que facilita la creación de rutas personalizadas para las diferentes funcionalidades de la aplicación.

(Stauffer, 2019) acota que “Laravel es un framework PHP potente y flexible. Cuenta con una próspera comunidad y un amplio ecosistema de herramientas, y como resultado está creciendo en atractivo y alcance. Además, la documentación de Laravel es exhaustiva y excelente. Actualmente Laravel ayuda a llevar sus ideas a la realidad sin desperdiciar código, utilizando modernos estándares de código, rodeado de una comunidad vibrante y un ecosistema de herramientas.”

Laravel se destaca por su potencia y flexibilidad como framework de desarrollo web. Este enfoque se refleja en su creciente atractivo y alcance en la comunidad de desarrolladores. El marco no solo ofrece una sintaxis clara y una estructura legible, sino que también proporciona un amplio ecosistema de herramientas y recursos que ayudan a los desarrolladores a materializar sus ideas de manera eficiente.

Laravel ha demostrado ser una herramienta confiable y efectiva para llevar a cabo proyectos web de diversos tamaños y complejidades, desde sitios web simples hasta aplicaciones empresariales de gran envergadura. En resumen, Laravel no solo se destaca por su elegante sintaxis y su seguridad, sino que también se ha convertido en un aliado invaluable para los desarrolladores, gracias a su próspera comunidad y su enfoque en la eficiencia y la calidad del desarrollo.

2.7. Base de Datos MySQL

Según (DuBois, 2008), “MySQL es un sistema de administración de bases de datos relacional (RDBMS). Se trata de un programa capaz de almacenar una enorme cantidad de datos de gran variedad y de distribuirlos para cubrir las necesidades de cualquier tipo de organización, desde pequeños establecimientos comerciales a grandes empresas y organismos administrativos.” Hoy en día MySQL es usado en su gran mayoría ya que posee una licencia de código abierto, adaptabilidad, rendimiento excelso y escalabilidad, lo que la hace adecuado para gestionar grandes volúmenes de datos y cargas de trabajo intensas.

MySQL ofrece diferentes ventajas, algunas de las cuales son:

- Código abierto: Podemos usarla sin incurrir a gastos de licencia, lo que lo hace asequible para una amplia gama de proyectos.
- Rendimiento: Ofrece un rendimiento elevado con altas cargas de trabajo, ofrece técnicas de indexación, optimización de consultas y caché eficiente.
- Escalabilidad: Maneja bases de datos pequeñas como grandes, ofrece escalabilidad vertical como horizontal.

- Compatibilidad: Es compatible con una variedad de lenguajes de programación, lo que facilita la integración en aplicaciones y sistemas existentes.
- Transacciones ACID: MySQL es capaz de manejar transacciones ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad), lo que garantiza la integridad y consistencia de los datos en aplicaciones que requieren operaciones transaccionales.

2.8. Lenguaje Unificado de Modelado (UML)

El lenguaje unificado de modelado es (Edraw, 2022): "una representación gráfica para el desarrollo y la ingeniería del software ya que nos ayudan a proporcionar una estructura general del software y el flujo de instrucciones." El uso de UML ayuda a los desarrolladores a comunicar de manera efectiva una representación del sistema que sea comprensible para personas sin experiencia en el área.

La elaboración de un Diagrama de Modelado Unificado (UML) se recomienda ampliamente debido a las múltiples ventajas que ofrece. En primer lugar, simplifica la comprensión de ideas y sistemas complejos, tanto para los desarrolladores como para los usuarios o clientes, transformando el código complejo en un diagrama visual que representa de manera clara la estructura, el comportamiento y las interacciones de los componentes del sistema.

3. CAPITULO III: DESARROLLO DEL PROYECTO

Este capítulo es donde se profundiza en la creación y construcción de la aplicación web destinada a la administración de cobros de cuotas de urbanizaciones. Este capítulo está dividido en varias fases, cada una de las cuales cumple un propósito específico en el ciclo de desarrollo de software.

3.1. Fase de Diseño

3.1.1. Requerimientos

3.1.1.1. Requerimientos Funcionales

Gestión de Usuarios: El aplicativo debe ofrecer la posibilidad de registro a los usuarios, creando una cuenta única para ingresar al mismo.

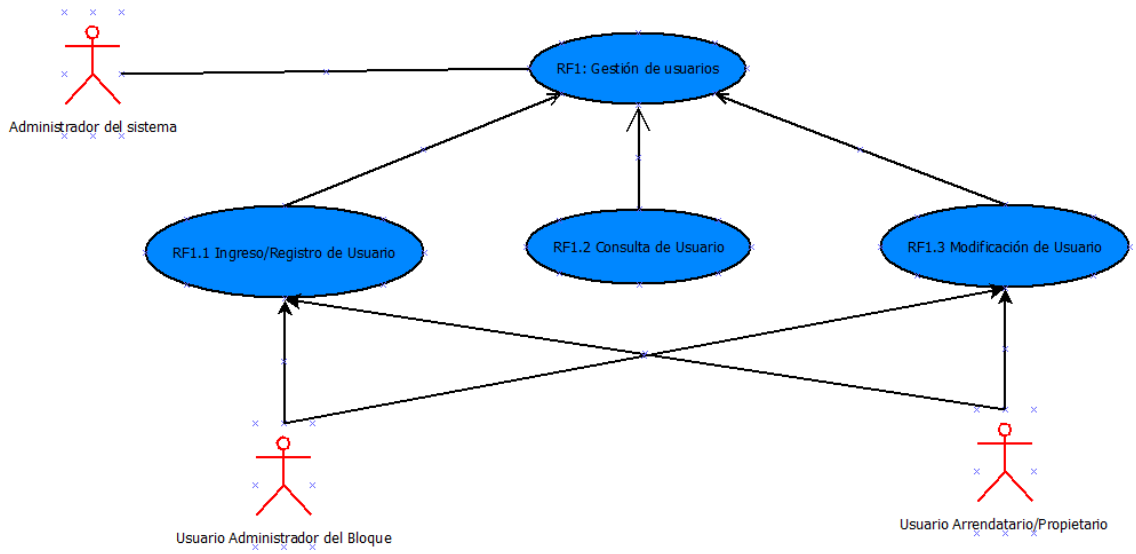

Ilustración 2: Diagrama de Siguiente Nivel de la Gestión de Usuarios (RF01).

- **Caso de Uso (RF01):**

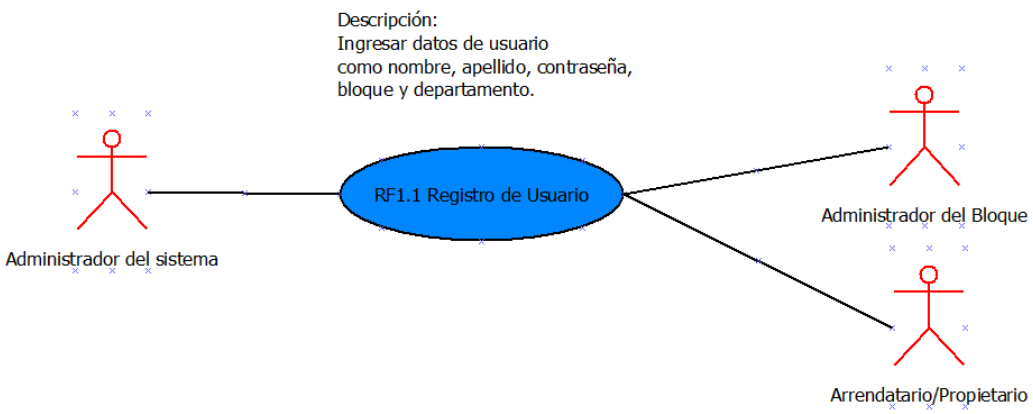

Ilustración 3: Caso de Uso RF1.1. Registro de Usuario.

Tabla 1: Caso de Uso RF1.1 - Registro de Usuario.

Caso de Uso	Registro de Usuario	**Identificador:** RF01
Actores	Usuario no registrado	
Tipo	Primario	
Referencias	Requerimientos relacionados: RF01 - Registro de Usuario Casos de uso relacionados: Ninguno	
Precondición	El usuario no debe estar registrado previamente en el aplicativo.	
Postcondición	El usuario crea una cuenta en el aplicativo y puede iniciar sesión con sus credenciales. El aplicativo almacena la información del usuario en la base de datos.	
Descripción	Este caso de uso permite a un usuario no registrado crear una cuenta en el aplicativo proporcionando información personal como nombre, correo electrónico, contraseña, bloque y departamento al que pertenece.	
Resumen	El usuario no registrado completa un formulario de registro con información personal, el aplicativo valida esta información y, si es válida, crea una cuenta para el usuario.	

Curso Normal

Tabla 2: Actividades del Caso de Uso RF1.1 - Registro de Usuario.

Nro.	Ejecutor	Paso o Actividad
1	Usuario no registrado	El usuario accede a la página principal del aplicativo
2	Usuario no registrado	El usuario accede a la página de registro
3	Aplicativo	El aplicativo muestra un formulario de registro con campos para nombre, correo electrónico, contraseña, bloque y departamento al que pertenece.
4	Usuario no registrado	El usuario completa el formulario, ingresando su información personal
5	Usuario no registrado	El usuario hace clic en el botón "Registrarse"
6	Aplicativo	El aplicativo valida la información proporcionada por el usuario (por ejemplo, verifica si el correo electrónico es único)
7	Aplicativo	Si la información es válida, el aplicativo crea una cuenta para el usuario y muestra un mensaje de registro exitoso.
8	Usuario Registrado	El usuario puede iniciar sesión en su nueva cuenta.

Cursos Alternos

Tabla 3: Acciones Alternas del Caso de Uso RF1.1 - Registro de Usuario

Nro.	Descripción de acciones alternas
6	**Información Incompleta o No válida:** Si el usuario proporciona información incompleta o no válida, el aplicativo muestra mensajes de error indicando los campos que requieren corrección.

- **Caso de Uso (RF02):**

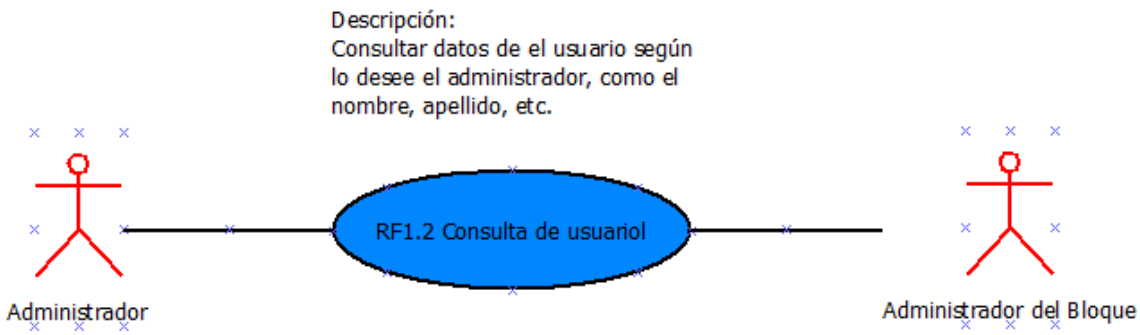

Ilustración 4: Caso de Uso RF1.2. Consulta de Usuario.

Tabla 4: Caso de Uso RF1.2 - Consulta de Usuario.

Caso de Uso	Consulta de Usuario	**Identificador:** RF03
Actores	Administrador de bloque	
Tipo	Primario	
Referencias	-	
Precondición	El administrador de bloque debe estar registrado en el aplicativo	
Postcondición	El administrador de bloque visualiza la información de los usuarios	
Descripción	Este caso de uso permite al administrador de bloque consultar información de los usuarios por bloque y departamento	
Resumen	El administrador de bloque accede a la información de los usuarios por bloque y departamento	

Curso Normal

Tabla 5: Actividades de Caso de Uso RF1.2 - Consulta de Usuario.

Nro.	Ejecutor	Paso o Actividad
1	Administrador de bloque	Inicia sesión
2	Administrador de bloque	Se dirige a la pestaña de Bloques
3	Administrador de bloque	Clickea en el apartado Departamento
4	Administrador de bloque	Clickea en el número de piso (del 1 al 5)
5	Administrador de bloque	Selecciona la letra del departamento (A, B o C)
6	Administrador de bloque	Visualiza la información de los usuarios con pagos pendientes en ese departamento

- **Caso de Uso (RF03):**

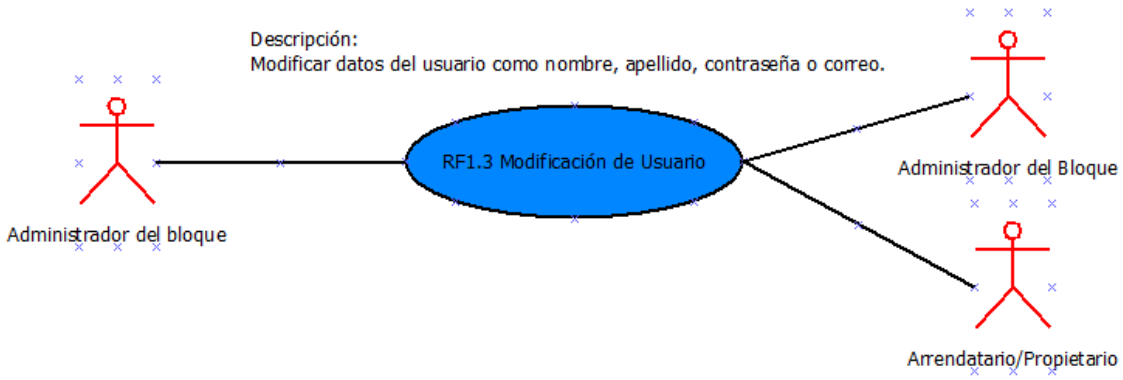

Ilustración 5: Caso de Uso RF1.3. Modificación de Usuario.

Tabla 6: Caso de Uso RF1.3 - Modificación de Usuario.

Caso de Uso	Modificación de Usuario	**Identificador:** RF02
Actores	Usuario registrado	
Tipo	Primario	

Referencias	Requerimientos relacionados: RF02 - Modificación de Usuario
Precondición	El usuario no debe estar registrado previamente en el aplicativo.
Postcondición	El usuario modifica su información personal
Descripción	Este caso de uso permite a un usuario registrado modificar su información en el aplicativo
Resumen	El usuario registrado modifica su información personal a través de la página de perfil

Curso Normal

Tabla 7: Actividades de Caso de Uso RF1.3 - Modificación de Usuario.

Nro.	Ejecutor	Paso o Actividad
1	Usuario registrado	Inicia sesión
2	Usuario no registrado	Se dirige a la pestaña de perfil
3	Usuario no registrado	Cambia su información (contraseña, nombre, apellido, correo electrónico, etc.)
4	Usuario registrado	Da clic en "Guardar"

Cursos Alternos

Tabla 8: Acciones Alternas de Caso de Uso RF1.3 - Modificación de Usuario.

Nro.	Descripción de acciones alternas
3	**Información Incompleta o No válida:** Si el usuario proporciona información incompleta o no válida, el aplicativo muestra mensajes de error indicando los campos que requieren corrección.

Gestión de Administración de Cobros: El aplicativo debe permitir el registro de cada pago realizado por un usuario

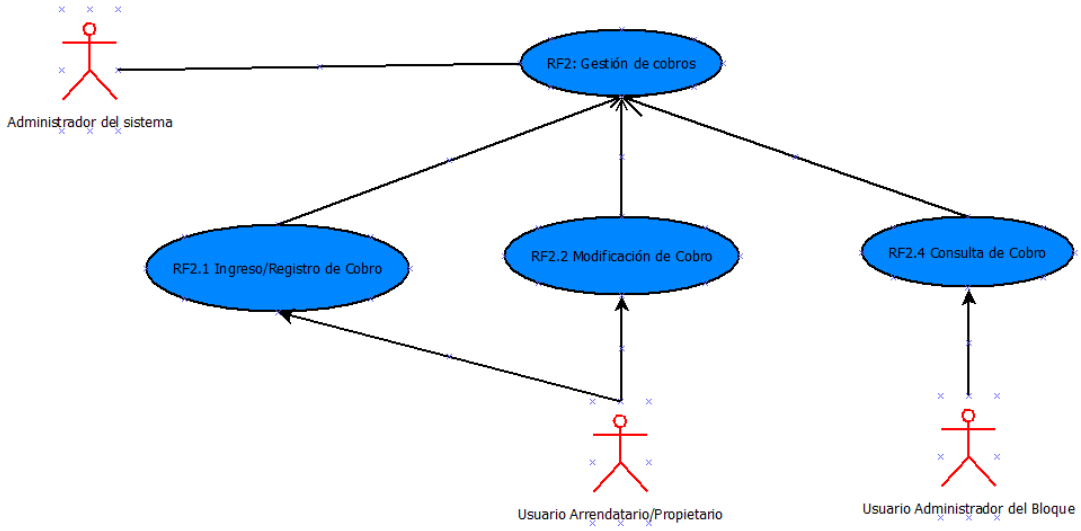

Ilustración 6: Diagrama de Siguiente Nivel de la Gestión de Cobros (RF02).

- **Caso de Uso (RF04):**

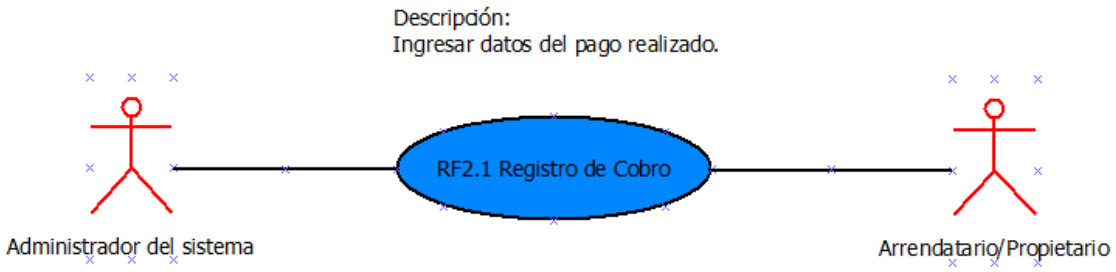

Ilustración 7: Caso de Uso RF2.1. Registro de cobro

Tabla 9: Caso de Uso RF2.1 - Registro de Cobro.

Caso de Uso	Registro de Cobro	**Identificador:** RF02
Actores	Usuario registrado	
Tipo	Primario	
Referencias	Requerimientos relacionados: RF02 - Registro de Cobro	
Precondición	El usuario debe estar registrado en el aplicativo como propietario o arrendatario.	

Postcondición	El usuario crea una un cobro para registrar su pago de cuota en el aplicativo. El aplicativo almacena la información del cobro en la base de datos.
Descripción	Este caso de uso permite a un usuario registrado crear un cobro en el aplicativo.
Resumen	El usuario registrado completa un formulario de registro de su pago con información personal.

Curso Normal

Tabla 10: Actividades de Caso de Uso RF2.1 - Registro de Cobro.

Nro.	Ejecutor	Paso o Actividad
1	Usuario Propietario o Arrendatario	El usuario accede a la página principal del aplicativo
2	Usuario Propietario o Arrendatario	El usuario accede a la página de login
3	Aplicativo	El aplicativo muestra un formulario de login con los campos de correo electrónico y contraseña.
4	Usuario Propietario o Arrendatario	El usuario completa el formulario, ingresando su información personal
5	Usuario Propietario o Arrendatario	El usuario hace clic en el botón "Iniciar Sesión"
6	Aplicativo	El aplicativo valida la información proporcionada por el usuario (por ejemplo, verifica si el correo electrónico es único)
7	Aplicativo	Si la información es válida, el aplicativo ingresa a la pestaña de usuario arrendatario/propietario.

8	Usuario Propietario o Arrendatario	El usuario puede iniciar sesión en su nueva cuenta.
9	Usuario Propietario o Arrendatario	El usuario ingresa a la pestaña de registrar pago
10	Usuario Propietario o Arrendatario	El usuario agrega el monto y señala el mes
11	Aplicativo	El usuario clickea en guardar registro.
12	Usuario Propietario o Arrendatario	El aplicativo valida la información
13	Usuario Propietario o Arrendatario	El aplicativo guarda el cobro en la base de datos.

Cursos Alternos

Tabla 11: Acciones Alternas de Caso de Uso RF2.1 - Registro de Cobro.

Nro.	Descripción de acciones alternas
7, 10	**Información Incompleta o No válida:** Si el usuario proporciona información incompleta o no válida, el aplicativo muestra mensajes de error indicando los campos que requieren corrección.

- **Caso de Uso (RF05):**

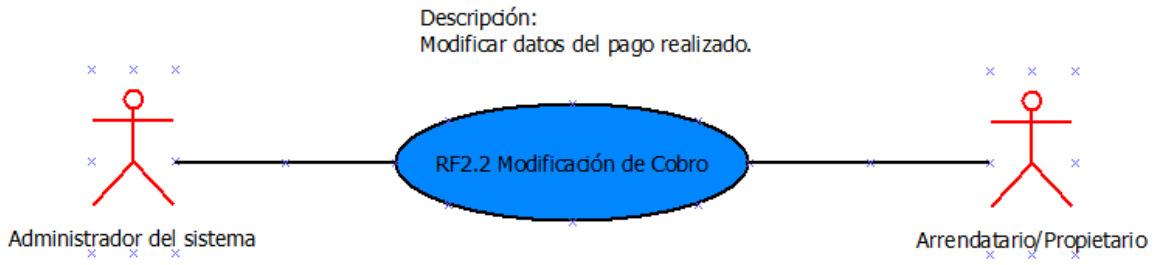

Ilustración 8: Caso de Uso RF2.2. Modificación de cobro

Tabla 12: Caso de Uso RF2.2 - Modificación de Cobro.

Caso de Uso	Modificación de Cobro	**Identificador:** RF05
Actores	Usuario registrado	

Tipo	Primario
Referencias	Requerimientos relacionados: RF04 - Modificación de Cobro
Precondición	El usuario debe estar registrado en el aplicativo
Postcondición	El usuario modifica la información de su pago
Descripción	Este caso de uso permite a un usuario registrado modificar los datos de su pago en el aplicativo
Resumen	El usuario registrado modifica el mes seleccionado para su pago o el monto a través de la página de cobros

Curso Normal

Tabla 13: Actividades de Caso de Uso RF2.2 - Modificación de Cobro.

Nro.	Ejecutor	Paso o Actividad
1	Usuario registrado	Inicia sesión
2	Usuario registrado	Se dirige a la pestaña de cobros
3	Usuario registrado	Cambia la información del pago (mes seleccionado, monto, etc.)
4	Usuario registrado	Da clic en "Guardar"

Cursos Alternos

Tabla 14: Acciones Alternas de Caso de Uso RF2.2 - Modificación de Cobro.

Nro.	Descripción de acciones alternas
3	**Información Incompleta o No válida:** Si el usuario proporciona información incompleta o no válida, el aplicativo muestra mensajes de error indicando los campos que requieren corrección.

- **Caso de Uso (RF06):**

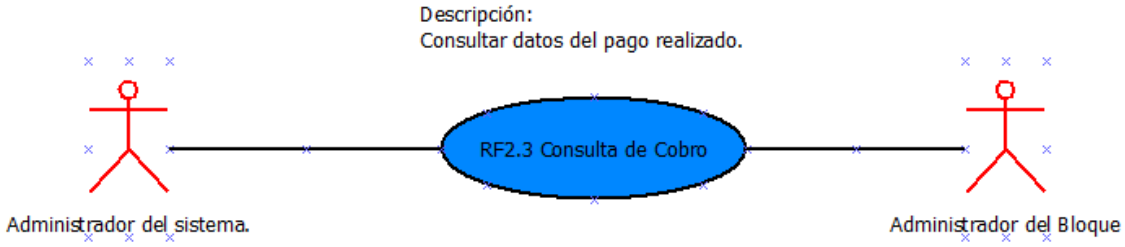

Ilustración 9: Caso de Uso RF2.3. Consulta de cobros

Tabla 15: Caso de Uso RF2.3 – Consulta de Cobro

Caso de Uso	Consulta de Cobro	**Identificador:** RF06
Actores	Administrador de bloque	
Tipo	Primario	
Referencias	-	
Precondición	El administrador de bloque debe estar registrado en el aplicativo	
Postcondición	El administrador de bloque visualiza la información de los cobros/pagos	
Descripción	Este caso de uso permite al administrador de bloque consultar los cobros/pagos por bloque y departamento	
Resumen	El administrador de bloque accede a la información de los cobros/pagos por bloque y departamento	

Curso Normal

Tabla 16: Actividades de Caso de Uso RF2.3 – Consulta de Cobro

Nro.	Ejecutor	Paso o Actividad
1	Administrador de bloque	Inicia sesión
2	Administrador de bloque	Se dirige a la pestaña de Departamentos
3	Administrador de bloque	Clickea en el apartado Departamento

4	Administrador de bloque	Clickea en el número de piso (del 1 al 5)
5	Administrador de bloque	Selecciona la letra del departamento (A, B o C)
6	Administrador de bloque	Visualiza la información de los cobros/pagos en ese departamento

Gestión de Reportes: El aplicativo debe ser capaz de emitir reportes mensuales y anuales de los pagos realizados por cada usuario.

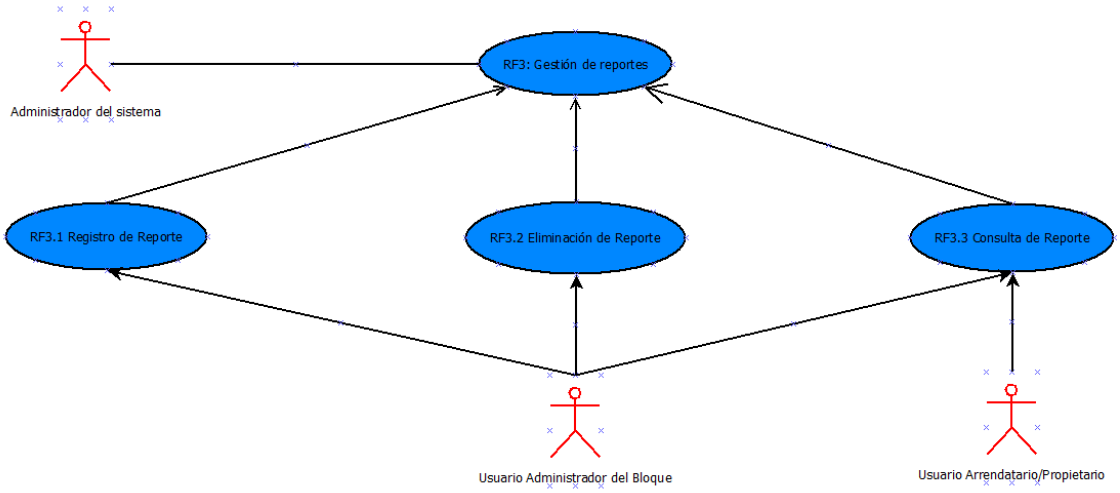

Ilustración 10: Diagrama de Siguiente Nivel de la Gestión de Reportes (RF03).

- **Caso de Uso (RF07):**

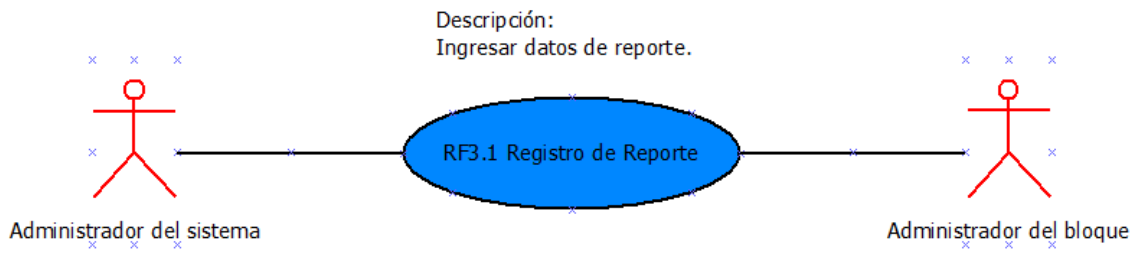

Ilustración 11: Caso de Uso RF3.1. Registro de Reporte

Tabla 17: Caso de Uso RF3.1 - Registro de Reporte.

Caso de Uso	Registro de Cobro	**Identificador:**

		RF02
Actores	Reporte registrado	
Tipo	Primario	
Referencias	N/A	
Precondición	El usuario debe estar registrado en el aplicativo como propietario o arrendatario.	
Postcondición	El usuario crea una un reporte. El aplicativo almacena la información del reporte en la base de datos.	
Descripción	Este caso de uso permite a un usuario registrado crear un reporte en el aplicativo.	
Resumen	El usuario registrado genera un nuevo reporte para su posterior visualización	

Curso Normal

Tabla 18: Actividades de Caso de Uso RF3.1 - Registro de Reporte.

Nro.	Ejecutor	Paso o Actividad
1	Usuario Propietario o Arrendatario	El usuario accede a la página principal del aplicativo
2	Usuario Propietario o Arrendatario	El usuario accede a la página de login
3	Aplicativo	El aplicativo muestra un formulario de login con los campos de correo electrónico y contraseña.
4	Usuario Propietario o Arrendatario	El usuario completa el formulario, ingresando su información personal
5	Usuario Propietario o Arrendatario	El usuario hace clic en el botón "Iniciar Sesión"
6	Aplicativo	El aplicativo valida la información proporcionada por el usuario (por

		ejemplo, verifica si el correo electrónico es único)
7	Aplicativo	Si la información es válida, el aplicativo ingresa a la pestaña de usuario arrendatario/propietario.
8	Usuario Propietario o Arrendatario	El usuario puede iniciar sesión en su nueva cuenta.
9	Usuario Propietario o Arrendatario	El usuario ingresa a la pestaña de registrar pago
10	Usuario Propietario o Arrendatario	El usuario agrega el monto y señala el mes
11	Aplicativo	El usuario clickea en guardar registro.
12	Usuario Propietario o Arrendatario	El aplicativo valida la información
13	Usuario Propietario o Arrendatario	El aplicativo guarda el cobro en la base de datos.

Cursos Alternos

Tabla 19: Acciones Alternas de Caso de Uso RF3.1 - Registro de Reporte.

Nro.	Descripción de acciones alternas
7, 10	**Información Incompleta o No válida:** Si el usuario proporciona información incompleta o no válida, el aplicativo muestra mensajes de error indicando los campos que requieren corrección.

- **Caso de Uso (RF08):**

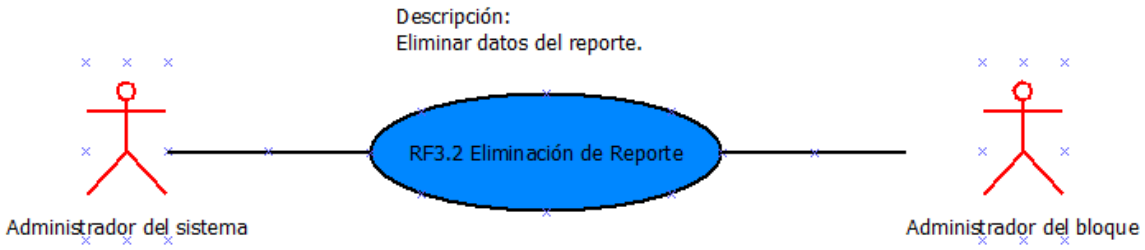

Ilustración 4: Caso de Uso RF3.3. Eliminación de Reporte

Tabla 20: Caso de Uso RF3.2 – Eliminación de Reporte

Caso de Uso	Eliminación de Reporte	**Identificador:** RF08
Actores	Administrador del bloque	
Tipo	Primario	
Referencias	Requerimientos relacionados: RF08 – Eliminación de Reporte	
Precondición	El administrador del bloque debe estar registrado en el aplicativo	
Postcondición	El reporte es eliminado	
Descripción	Este caso de uso permite al administrador del bloque eliminar un reporte en el aplicativo	
Resumen	El administrador del bloque elimina un reporte erróneo o que ha pasado su tiempo de vida útil a través de la página de reportes	

Curso Normal

Tabla 21: Actividades de Caso de Uso RF3.2 – Eliminación de Reporte

Nro.	**Ejecutor**	**Paso o Actividad**
1	Administrador del bloque	Inicia sesión
2	Administrador del bloque	Se dirige a la pestaña de reportes
3	Administrador del bloque	Elimina el reporte
4	Administrador del bloque	Da clic en "Confirmar"

- **Caso de Uso (RF09):**

Ilustración 5: Caso de Uso RF3.4. Consulta de Reporte

Tabla 22: Caso de Uso RF3.3 – Consulta de Reporte

Caso de Uso	Consulta de Reporte	**Identificador:** RF09
Actores	Propietario/Arrendatario	
Tipo	Primario	
Referencias	-	
Precondición	El propietario o arrendatario debe estar registrado en el aplicativo	
Postcondición	El usuario visualiza el reporte	
Descripción	Este caso de uso permite al propietario o arrendatario consultar reportes mensuales o anuales	
Resumen	El propietario o arrendatario accede a la información de reportes mensuales o anuales a través de la página de reportes	

Curso Normal

Tabla 23: Actividades de Caso de Uso RF3.3 – Consulta de Reporte

Nro.	**Ejecutor**	**Paso o Actividad**
1	Propietario/Arrendatario	Inicia sesión
2	Propietario/Arrendatario	Se dirige a la pestaña de Reportes
3	Propietario/Arrendatario	Clickea en el apartado de meses, o años si está disponible

4	Propietario/Arrendatario	Clickea en "Visualizar Reporte"
5	Propietario/Arrendatario	Visualiza la información del reporte

Cursos Alternos

Tabla 24: Acciones Alternas de Caso de Uso RF3.3 – Consulta de Reporte

Nro.	Descripción de acciones alternas
4	**Información incompleta:** Si no existe información suficiente para generar un reporte, dirigirse a los propietarios/arrendatarios para realizar pagos.

3.1.1.2. Requerimientos No Funcionales

Interfaz de Usuario Intuitiva:

- Pantallas de inicio de sesión y registro.
- Panel de administración para la gestión de cobros.
- Perfiles de usuario con información personal.

Seguridad:

- Protección de contraseñas mediante hash.
- Restricciones de acceso basadas en roles de usuario.

Rendimiento:

- Respuesta rápida a las solicitudes de los usuarios.
- Manejo eficiente de grandes conjuntos de datos.

Disponibilidad:

- Alta disponibilidad para garantizar que la aplicación esté siempre disponible.

Compatibilidad:

- Compatibilidad con navegadores web modernos.
- Soporte para dispositivos móviles.

Mantenibilidad:

- Código limpio y bien documentado.
- Facilidad para realizar actualizaciones y mejoras.

3.1.1.3. Herramientas por utilizar

XAMPP: XAMPP es un paquete de software que proporciona un entorno de servidor web local para el desarrollo y prueba de aplicaciones web. Incluye componentes como Apache (servidor web), MySQL (base de datos), PHP y Perl, lo que lo hace útil para desarrollar y depurar aplicaciones web antes de desplegarlas en un servidor en producción.

Visual Studio Code: Visual Studio Code (VS Code) es un editor de código fuente altamente personalizable y de código abierto desarrollado por Microsoft. Es ampliamente utilizado por desarrolladores web y ofrece soporte para una variedad de lenguajes de programación y extensiones que facilitan la programación.

Laravel: Laravel es un popular marco de desarrollo de aplicaciones web en PHP. Ofrece una sintaxis elegante y herramientas poderosas para simplificar el desarrollo de aplicaciones web. Laravel se utiliza comúnmente para la creación de aplicaciones web robustas y seguras.

Laravel Breeze: Es un kit de inicio oficial proporcionado por Laravel. Está diseñado para simplificar la configuración inicial de autenticación en las aplicaciones Laravel. Breeze ofrece una manera rápida de configurar la autenticación básica, como el registro, inicio de sesión, restablecimiento de contraseña, etc. En resumen, según (DesarrolloWeb, 2023), "Laravel Breeze es un scaffolding elemental de todo el proceso de login y registro de usuarios. Que quitará mucho trabajo en aplicaciones nuevas que requieran usuarios y ayudará a comenzar con un conjunto de archivos un poco más amplio".

Composer: Composer es una herramienta de administración de dependencias en PHP que permite la instalación y gestión de bibliotecas y paquetes de código externo en proyectos de PHP.

Artisan: Artisan es la interfaz de línea de comandos (CLI) incluida en Laravel. Permite realizar diversas tareas de desarrollo, como la generación de controladores, modelos, migraciones y ejecución de comandos personalizados para la administración de la aplicación Laravel.

MySQL: MySQL es un sistema de gestión de bases de datos relacional de código abierto ampliamente utilizado. MySQL servirá como el sistema de gestión de bases de datos para almacenar y administrar los datos relacionados con los usuarios, los cobros de cuotas y otros aspectos de la aplicación. MySQL es conocido por su velocidad, confiabilidad y capacidad de escalabilidad, lo que lo hace una elección sólida para aplicaciones web.

PowerDesigner: PowerDesigner es una herramienta de modelado de datos y arquitectura empresarial desarrollada por SAP. Permite diseñar y documentar la estructura de la base de datos, así como modelar los procesos de negocio y las relaciones entre los elementos del sistema. PowerDesigner sería útil para diseñar y visualizar la estructura de la base de datos que respalda la aplicación de administración de cobros de cuotas de urbanizaciones. Esto facilita la comprensión y la colaboración en el diseño de la base de datos y el flujo de datos en la aplicación. PowerDesigner también puede generar esquemas de bases de datos que se pueden implementar en sistemas de gestión de bases de datos como MySQL.

3.1.1.4. Casos de Uso

- **Diagrama General de Casos de Uso:** En este diagrama se puede observar cómo los actores interactúan con los diferentes casos de uso en el aplicativo de gestión de cobros de departamentos. Cada caso de uso (RF) representa una funcionalidad específica que puede realizarse en el sistema.

  En este diagrama:

  El "Administrador del Aplicativo" se conecta a todos los casos de uso, lo que incluye ingresar al sistema, gestionar usuarios, cobros, reportes y salir del sistema.

  El "Administrador de Bloque" se conecta a todos los casos de uso excepto la gestión de usuarios (RF1). Esto significa que el Administrador de Bloque puede realizar acciones relacionadas con el mantenimiento de bloques, la gestión de departamentos, el pago de cuotas y la generación de reportes.

  El "Usuario Propietario/Arrendatario de Departamento" también se conecta a todos los casos de uso excepto la gestión de usuarios (RF1). Esto implica que el Usuario Propietario/Arrendatario puede interactuar con el sistema para gestionar su departamento, pagar cuotas y generar reportes de recaudación.

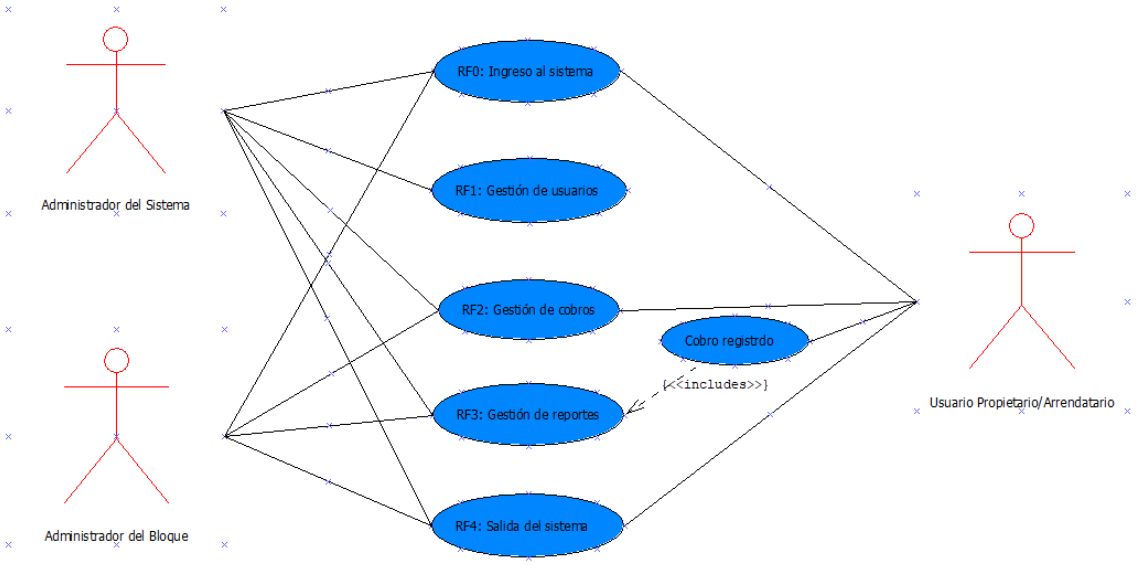

Ilustración 15: Diagrama General de Casos de Uso

3.2. Fase de Planificación

3.2.1. Primera Iteración

3.2.1.1. Requerimientos Primera Iteración

- **Base de Datos:** El sistema parte de la base de datos creada a partir de las tablas: departamentos y bloques, lo que permite crear como pagos y generar reportes.
- **Configuración del Proyecto:** El sistema debe configurarse inicialmente en los entornos específicos para su correcto funcionamiento.
- **Departamentos y Bloques:** El sistema permite asociar un departamento de un bloque en específico hacia un arrendatario o propietario. Un departamento y un bloque en el contexto de una urbanización corresponde a una sección o porción de un conjunto residencial que son la base de este proyecto.

3.2.1.2. Tareas Primera Iteración

El proceso de desarrollo de la aplicación se inicia con la configuración del proyecto en el framework Laravel y la creación de un repositorio Git para el control de versiones. Esta base proporciona la estructura necesaria para el desarrollo ordenado de la aplicación.

Posteriormente se procede a la instalación de PHP mediante cualquier distribución de apache, la que yo elegí fue XAMPP, luego, para el uso y modelado de la base de datos se instaló MySQL, además para el manejo de paquetes de PHP se instaló Composer, esto para

poder realizar la correcta creación de un proyecto; una de las últimas herramientas que se necesitaron fue Node.js, esto con la finalidad de poder compilar el código Javascript y CSS. Para poder escribir mi código se decidió por el IDE Visual Studio Code, por lo que se instaló esta herramienta. Por último se procedió con la instalación de Artisan, esto con la finalidad de realizar diversas tareas como crear controladores, modelos, migraciones de base de datos, ejecutar migraciones o sembrar datos de prueba.

En la primera fase del desarrollo, se enfoca en la creación de interfaces que permitan visualizar información relacionada con los departamentos, ya sea por parte de los propietarios o arrendatarios. Se establecen rutas de navegación para acceder a estas interfaces y ofrecer una experiencia de usuario intuitiva. Además, se implementa un sistema de autenticación (login) que permite a los usuarios con roles de administrador acceder al sistema de gestión.

Por último, los requisitos de bloques y departamentos se integran en la base de datos. Se crean tablas específicas para representar estas entidades, lo que facilita el almacenamiento y gestión de información crucial para el funcionamiento del sistema. Este enfoque estructurado garantiza que la aplicación cumpla con los requerimientos definidos, ofreciendo un sistema funcional y eficiente para la gestión de bloques, departamentos y usuarios.

3.2.2. Segunda Iteración

3.2.2.1. Requerimientos Segunda Iteración

- **Usuarios:** El aplicativo tendrá la posibilidad de permitir a los usuarios registrarse y crear una cuenta de usuario única para acceder al sistema, con roles definidos como administrador de bloque y arrendatario o propietario.

3.2.2.2. Tareas Segunda Iteración

El proceso de inicio de sesión en la aplicación es crucial para la gestión de usuarios y roles. Los roles asignados varían según el tipo de usuario: administrador del bloque o arrendatario/propietario. La autenticación se realiza mediante credenciales proporcionadas por el administrador del sistema, y estas credenciales son exclusivas para los 33 administradores de bloque. Cada administrador del bloque tiene acceso a funciones específicas relacionadas con la gestión de bloques y departamentos.

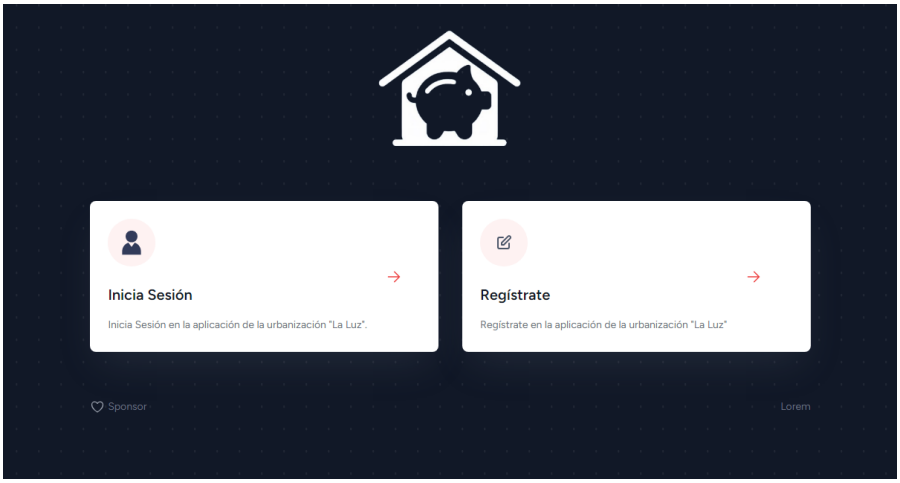

Ilustración 14: Pantalla de Inicio del Aplicativo

Por otro lado, los usuarios arrendatarios/propietarios deben ingresar información personal, como nombres, departamento y bloque de residencia, al registrarse en la aplicación. Cada usuario solo puede elegir una combinación única de departamento y bloque. Esto garantiza un acceso seguro y restringe a cada usuario a su área de residencia específica.

Ilustración 6: Pantalla Inicio de Sesión

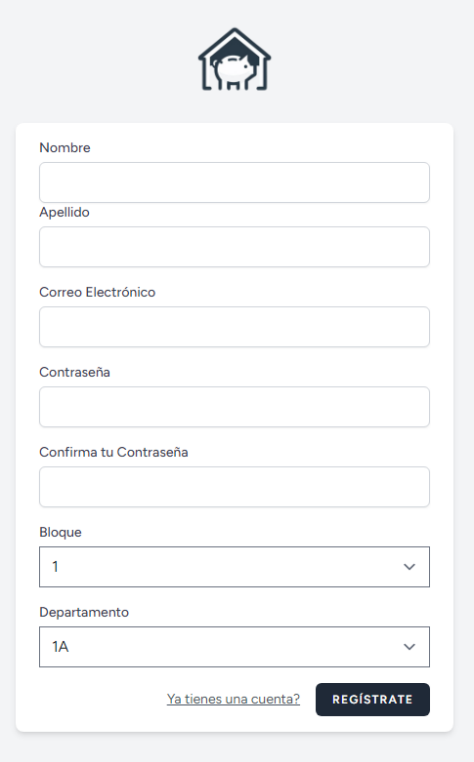

Ilustración 7: Pantalla Registro de Usuario

La funcionalidad de inicio de sesión se basa en la autenticación de usuarios y en la asignación de roles para determinar las acciones permitidas dentro del sistema. Este proceso es fundamental para mantener la privacidad y la seguridad de la información en la aplicación y para asegurar que cada usuario acceda solo a los recursos autorizados.

3.2.3. Tercera Iteración

3.2.3.1. Requerimientos Tercera Iteración

- **Reportes**: El aplicativo debe ser capaz de emitir reportes mensuales o anuales.

3.2.3.2. Tareas Tercera Iteración

La generación de reportes mensuales o anuales por bloque es una característica clave en la aplicación y está diseñada para proporcionar información financiera importante. El administrador de cada bloque es el usuario autorizado para generar estos informes. Su rol incluye la responsabilidad de supervisar y mantener el estado financiero de su bloque específico. A través de esta funcionalidad, el administrador puede obtener información detallada sobre quién ha realizado pagos y cuánto dinero se ha recaudado durante un período de tiempo específico.

Por otro lado, los usuarios propietarios/arrendatarios también tienen acceso a esta funcionalidad, pero su capacidad se limita a la visualización y descarga de los informes en formato PDF. Esto les permite mantener un registro de sus propios pagos y asegurarse de que sus finanzas estén en orden. La función de descarga en PDF facilita el almacenamiento y el acceso a los informes para su referencia personal.

La característica de generación de informes mensuales o anuales por bloque es una parte esencial de la aplicación que sirve tanto a los administradores de bloques como a los usuarios propietarios/arrendatarios. Proporciona información financiera crítica para tomar decisiones informadas y mantener un registro detallado de los pagos y la recaudación de dinero.

3.3. Fase de Codificación

3.3.1. Código

En este sistema, se ha utilizado Laravel, un potente marco de desarrollo de aplicaciones web en PHP, para crear una plataforma de gestión de departamentos y bloques. El sistema está diseñado para permitir a los usuarios, tanto administradores como propietarios/arrendatarios, gestionar y mantener sus departamentos y bloques de manera eficiente. Laravel proporciona un entorno robusto y seguro para desarrollar esta aplicación y facilita la implementación de funciones clave como inicio de sesión, administración de usuarios, generación de reportes y más.

- **Rutas:** En Laravel, las rutas definen la URL y la lógica asociada con ella. Las rutas pueden ejecutar funciones anónimas o apuntar a métodos en controladores.
- **Controladores:** Los controladores son clases que manejan la lógica de la aplicación. Organizan la lógica relacionada en métodos, como mostrar una vista o guardar datos en la base de datos.
- **Vistas:** Las vistas son archivos que definen la interfaz de usuario de tu aplicación. Laravel utiliza el motor de plantillas Blade para crear vistas dinámicas. Los controladores pueden pasar datos a las vistas para mostrar información.
- **Migraciones:** Las migraciones se utilizan para crear y modificar la estructura de la base de datos. Puedes definir esquemas de tablas y ejecutar migraciones para aplicar estos cambios en la base de datos.

- **Middlewares:** Los middlewares son filtros que se ejecutan antes o después de una solicitud HTTP. Se utilizan para realizar tareas como la autenticación, autorización o el registro de actividades antes de llegar al controlador.
- **Autenticación:** Laravel proporciona un sistema de autenticación incorporado con métodos para registrar, iniciar sesión y gestionar usuarios. Se pueden personalizar para satisfacer las necesidades específicas de la aplicación.
- **APIs:** Laravel facilita la creación de APIs RESTful utilizando rutas y controladores. Las respuestas pueden estar en formato JSON para la comunicación con aplicaciones frontend o servicios externos.
- **Artisan:** Artisan es la herramienta de línea de comandos de Laravel que simplifica tareas como la creación de controladores, modelos, migraciones, entre otros. Ofrece comandos útiles para la gestión de la aplicación.
- **Blade:** Blade es el motor de plantillas incorporado de Laravel. Permite definir estructuras de vista reutilizables y control de flujo dentro de tus vistas. Facilita la creación de vistas dinámicas.
- **Base de Datos:** Laravel es compatible con varias bases de datos, incluyendo MySQL y SQLite. Facilita la implementación de autorización y autenticación.

3.4. Arquitectura de Software

He optado por una arquitectura de software basada en el Modelo-Vista-Controlador (MVC), lo que facilita la organización y desarrollo de la aplicación. En esta arquitectura, el Modelo se encarga de definir la estructura de la base de datos, lo que es crucial para representar usuarios, bloques, departamentos, cobros y reportes.

Por otro lado, la capa de Vista se ha construido utilizando el motor de plantillas Blade de Laravel, permitiendo la creación de interfaces de usuario dinámicas. Las vistas se emplean para mostrar datos relevantes a los usuarios, como páginas web, formularios de registro y reportes generados. Esta flexibilidad en la presentación de datos es esencial para brindar una experiencia amigable y efectiva.

Los Controladores han resultado fundamentales para gestionar la lógica de la aplicación. Estos componentes actúan como intermediarios entre las rutas y las vistas, garantizando que las solicitudes HTTP se procesen correctamente. Así, he implementado la autenticación, la gestión de usuarios, la generación de reportes y la visualización de

departamentos de manera ordenada y eficiente, cumpliendo con los requerimientos de mi proyecto de gestión de propiedades.

3.5. Pruebas de la Aplicación Web

3.5.1. Pruebas de aceptación

Las pruebas de aceptación se realizan para validar que el software cumpla con los requisitos del cliente y para asegurar que nuevas funcionalidades del sistema no afecten a las existentes. Estas pruebas son esenciales para garantizar la calidad del software y la satisfacción del cliente, ya que permiten verificar que el sistema funciona según lo esperado y que cumple con las expectativas del usuario final.

Caso RF01_P01

Tabla 25: PruebaAceptación Caso de Uso RF01_P01

Pruebas de Aceptación	
Código: RF01_P01	**Requerimiento:** R01
Nombre: Registro de Usuario	
Descripción: El sistema tendrá un formulario de registro de usuario.	
Condiciones: N/A	
Pasos: 1. El usuario accede a la página de registro desde el menú principal 2. El usuario completa el formulario, ingresando su información personal 3. Hacer click en el botón de registrarse	
Resultados previstos: El usuario se registrará exitosamente y será redirigido a la pantalla principal del menú de usuarios propietarios o arrendatarios.	
Evaluación: Prueba completada satisfactoriamente	

Caso RF04_P02

Tabla 26: Prueba de Aceptación Caso de Uso RF04_P02

Pruebas de Aceptación	
Código: RF04_P02	**Requerimiento:** R04
Nombre: Registro de Cobro	
Descripción: El sistema tendrá un módulo para la creación de un cobro para un usuario arrendatario/propietario	

Condiciones: N/A
Pasos: 1. El usuario inicia sesión 2. El usuario ingresa al apartado de ingreso de cobro 3. El usuario genera un nuevo cobro según el mes 4. El usuario clickea en guardar para generar un nuevo cobro
Resultados previstos: El usuario registrará exitosamente el cobro.
Evaluación: Prueba completada satisfactoriamente

Caso RF07_P03

Tabla 27: Prueba de Aceptación Caso de Uso RF07_P03

Pruebas de Aceptación	
Código: RF07_P03	**Requerimiento:** R07
Nombre: Registro de Reporte	
Descripción: El sistema tendrá un módulo para la creación de un reporte para un usuario administrador de bloque	
Condiciones: N/A	
Pasos: 1. El usuario accede a la página de reportes 2. El usuario genera un reporte según el mes que desee o el año 3. El usuario propietario/arrendatario visualizará el reporte	
Resultados previstos: El administrador de bloque generará satisfactoriamente un reporte para posteriormente ser visualizado por el arrendatario/propietario.	
Evaluación: Prueba completada satisfactoriamente	

Caso RF09_P04

Tabla 28: Prueba de Aceptación Caso de Uso RF09_P04

Pruebas de Aceptación	
Código: RF09_P04	**Requerimiento:** R09
Nombre: Consulta de Reporte	
Descripción: El sistema tendrá una funcionalidad de generación de reportes por mes y año	

Condiciones: N/A
Pasos: 1. El usuario accede a la página de inicio de sesión 2. El usuario accede al módulo de reportes 3. El usuario elige los reportes disponibles 4. El usuario visuliza el reporte elegido
Resultados previstos: El usuario visualizará los reportes correctamente en pantalla
Evaluación: Prueba completada satisfactoriamente

CAPITULO IV: CONCLUSIONES Y RECOMENDACIONES

4.1. Conclusiones

El alcance de este proyecto se centra en el desarrollo completo de un aplicativo web para la gestión de cobros de cuotas de departamentos en urbanizaciones. Se ha trabajado exhaustivamente en la implementación de funciones clave, como el registro de pagos, validación de información y generación de informes, con el objetivo fundamental de agilizar los procesos de cobro y mejorar la eficiencia en la gestión financiera de las urbanizaciones. Este enfoque integral abarca la automatización de tareas relacionadas con la administración de bloques, la gestión de departamentos y la emisión de reportes, proporcionando a los usuarios herramientas efectivas para un control más efectivo de las cuotas.

Durante la fase de diseño, se estableció la estructura de la base de datos y se detallaron los diferentes casos de uso del sistema. La planificación y diseño cuidadosos han sentado las bases para un desarrollo coherente y una implementación exitosa de las funcionalidades requeridas. La identificación y comprensión detallada de los procesos involucrados en la gestión de cobros han sido elementos fundamentales para la creación de un sistema que aborda de manera efectiva las necesidades específicas, como la validación de información y la generación precisa de informes financieros.

La elección de la arquitectura Modelo-Vista-Controlador (MVC) en la metodología de desarrollo refleja nuestra preocupación por la organización y estructuración eficiente del proyecto. Esta metodología ágil ha facilitado la implementación y gestión de las funcionalidades del aplicativo. Asimismo, la elección acertada de Laravel como marco de desarrollo ha proporcionado un entorno robusto para el desarrollo web en PHP, simplificando tareas clave como la autenticación, gestión de bases de datos y creación de interfaces dinámicas.

La implementación de roles de usuario (administrador de bloque, propietario/arrendatario) y un sistema de autenticación ha contribuido significativamente a la seguridad y personalización del sistema. Los usuarios tienen

acceso a funciones específicas según sus roles, fortaleciendo la integridad y eficacia del aplicativo.

4.2. Recomendaciones

La aplicación debe ser mantenida regularmente para abordar posibles problemas, implementar actualizaciones de seguridad y agregar nuevas funcionalidades según las necesidades cambiantes. Un plan de mantenimiento continuo garantizará la longevidad y eficacia del aplicativo.

Establecer un canal eficiente para la recepción y respuesta a problemas de los usuarios. Un sistema de soporte técnico o una sección de preguntas frecuentes pueden ser útiles para abordar rápidamente inquietudes y garantizar la satisfacción del usuario.

Considerar la posibilidad de escalabilidad es esencial para acomodar un posible aumento en el número de usuarios o la cantidad de datos. La arquitectura y el diseño deben permitir una expansión sin comprometer el rendimiento del aplicativo.

Asegurar la integridad y confidencialidad de los datos es crucial. Se recomienda implementar prácticas robustas de seguridad, como el cifrado de contraseñas, el uso de HTTPS, y la gestión segura de sesiones para proteger la información sensible de los usuarios.

Se recomienda establecer un plan de mantenimiento regular para abordar problemas potenciales, implementar actualizaciones de seguridad y agregar nuevas funciones. Esto garantizará la longevidad y eficacia del aplicativo a lo largo del tiempo.

Establecer un canal eficiente para la recepción y respuesta a problemas de los usuarios es esencial. Considerar la implementación de un sistema de soporte técnico o una sección de preguntas frecuentes para abordar rápidamente inquietudes y garantizar la satisfacción del usuario.

Anticipar la posibilidad de un aumento en el número de usuarios o datos es crucial. La arquitectura y diseño deben permitir una expansión sin comprometer el rendimiento. Además, se recomienda implementar prácticas robustas de seguridad, como el cifrado de contraseñas y el uso de HTTPS, para proteger la información sensible de los usuarios.

BIBLIOGRAFÍA

DesarrolloWeb. (25 de Agosto de 2023). *DesarrolloWeb*. Obtenido de https://desarrolloweb.com/articulos/laravel-breeze

DuBois, P. (2008). *La Biblia de MySQ.* Anaya Multimedia.

Edraw. (2022). *Edraw*. Obtenido de https://www.edrawsoft.com/es/uml/

Fernández Escribano, G. (09 de 12 de 2009). Introducción a Extreme Programming. *Actores y sus responsabilidades, II*. Madrid, Castilla La-Mancha, España.

Hills, H., & Klint, P. (2017). Enabling PHP software engineering research in Rascal. En H. Hills, & P. Klint, *Science of Computer Programming* (págs. 37-46).

Mestres, A. F. (2015). *Introducción al frontend y backend.* Barcelona: Universitat Oberta de Catalunya.

Oviedo, M. (2013). La gestión de las cuotas de urbanización en el sistema de cooperación. *Práctica urbanística: Revista mensual de urbanismo*, 24-39.

Pérez, O. A. (2011). Cuatro enfoques metodológicos para el desarrollo de Software RUP – MSF – XP - SCRUM. En O. A. Pérez, *Investigación de Metodologías Ágiles* (pág. 72). Quito, Pichincha, Ecuador: Inventum.

Quintana, J. (2023). *Instituto Cibertec | Carreras técnicas y formación continua*. Obtenido de ¿Qué es back-end y front-end?: https://www.cibertec.edu.pe/noticias/que-es-back-end-y-front-end/

Stauffer, M. (2019). *Laravel - Up and Running.* O'Reilly Media.

GLOSARIO

A

APIs: Interfaz de programación de aplicaciones que permite la comunicación entre diferentes sistemas.

B

Back-end: Parte de una aplicación que maneja la lógica y la gestión de datos detrás de escena.

Blade: Motor de plantillas incorporado en Laravel para la creación de vistas dinámicas.

C

Cifrado de contraseñas: Método de seguridad que transforma las contraseñas en una forma ilegible para proteger la información sensible.

Composer: Herramienta de administración de dependencias en PHP que permite la instalación y gestión de bibliotecas y paquetes de código externo.

Controladores: Clases que manejan la lógica de la aplicación en Laravel.

E

Escalabilidad: Capacidad del sistema para manejar un aumento en el número de usuarios o la cantidad de datos sin comprometer el rendimiento.

Extreme Programming (XP): Metodología de desarrollo ágil que se centra en la entrega rápida y continua de software de alta calidad.

H

HTTPS: Protocolo seguro de transferencia de hipertexto que garantiza la seguridad de la comunicación en la web.

HTTP: Protocolo de transferencia de hipertexto utilizado para la comunicación en la web.

L

Laravel Breeze: Kit de inicio oficial proporcionado por Laravel para simplificar la configuración inicial de autenticación en aplicaciones Laravel.

M

Middlewares: Filtros que se ejecutan antes o después de una solicitud HTTP en Laravel.

Modelo-Vista-Controlador (MVC): Arquitectura de software que organiza la aplicación en tres componentes principales: el modelo define la estructura de la base de datos, la vista se encarga de la interfaz de usuario, y el controlador gestiona la lógica de la aplicación.

Migraciones: Utilizadas en Laravel para crear y modificar la estructura de la base de datos.

MySQL: Sistema de gestión de bases de datos relacional de código abierto, conocido por su velocidad, confiabilidad y capacidad de escalabilidad.

R

Rutas: En Laravel, definen la URL y la lógica asociada con ella.

S

Soporte técnico: Sistema para la recepción y respuesta a problemas de los usuarios, puede incluir una sección de preguntas frecuentes.

V

Visual Studio Code (VS Code): Editor de código fuente altamente personalizable y de código abierto desarrollado por Microsoft.

W

Widget: Pequeño programa o aplicación que facilita el acceso a las funciones más utilizadas de un dispositivo.