Informe de análisis
Compilatio Magister+ | ZEGEL-PE

SISWEB-JASS_VF

ID : dc82d7e9327f454498026644f8104e7111f57c1b

3%
Textos sospechosos

Nombre del fichero : SISWEB-JASS_VF.txt
Tamaño del archivo original : 4,98 MB
Número de palabras : 17.539

Depositante : Jerson Sanchez Ramos
Fecha de depósito : 18 de septiembre de 2026
Tipo de carga : interface
fecha de fin de análisis : 18 de septiembre de 2026

Resumen (sección 1/3)

Localización de los textos sospechosos en el documento :



Incluido en el porcentaje de textos sospechosos :

Similitudes — **3%**

Sintáctica 3% | Semántica <1%

Pasajes con similitudes a fuentes encontradas en diferentes colecciones.



Detección de IA — **27%**

Textos estilísticamente próximos a un texto generado por una IA.
Este índice es un indicador y no una prueba. Comprueba con el autor si domina los conocimientos mencionados en el documento.



Idiomas no reconocidos — **25%**

Pasajes en los que parte del vocabulario utilizado no forma parte del diccionario de la lengua.
Puede tratarse de un intento del autor de modificar el texto para evitar ser detectado.



No incluido en el porcentaje de textos sospechosos :

Textos entre comillas — **<1%**

Pasajes entre comillas, a menudo indicativos de una cita.

Similitudes

3%

Pasajes con similitudes a fuentes encontradas en diferentes colecciones.



Fuente principal detectada

N°		Descripciones	Similitudes	Ubicaciones
1		Equipo 7 - Lazaro #2d5653 Detección semántica Viene de de mi grupo	2%	
2		¿Qué es el Back End y Front End? ⚡ nestrategia.com/desarrollo-web-back-end-front-end/	<1%	
4		Planning Poker: metodología Scrum para una... Detección semántica www.appvizer.es/revista/organizacion-planificacion/gestion-proyectos/pla...	<1%	

Fuente con similitudes fortuitas

N°		Descripciones	Similitudes	Ubicaciones
3		Documento de otro usuario #320ed3 Detección semántica Viene de de otro grupo	<1%	
5		Documento de otro usuario #02cac3 Viene de de otro grupo	<1%	
6		EPOC: El impacto de la Actividad Física en pacientes sielbleu.es/el-impacto-de-la-actividad-fisica-en-pacientes-con-la-enfermeda...	<1%	
7		Documento de otro usuario #fdd010 Viene de de otro grupo	<1%	

Fuente mencionada (sin similitudes detectadas)

N°		Descripciones
1		https://opencontent.upct.es/f0fc64e3b2d34f3a872a841a9c37b249/3f9bb933ba6440e59b0db2ecf6
2		https://www.lifeder.com/factibilidad-tecnica/
3		https://www.shopify.com/blog/feasibility-study
4		https://arxiv.org/abs/2103.15268
5		https://angular.dev
6		https://ittude-agile.com
7		https://www.nestrategia.com
8		https://asana.com/es/resources/swot-analysis

N°		Descripciones
9		https://www.arrobasolutions.com/metodologia-scrum/
10		https://blog.buhoos.com/artefactos-de-scrum/

1

INSTITUTO DE EDUCACIÓN SUPERIOR PRIVADO “IDAT" PROGRAMA DE ESTUDIOS EN DESARROLLO

DE SISTEMAS DE INFORMACIÓN

DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA WEB PARA OPTIMIZAR LA GESTIÓN DE COBROS Y REPORTES POR SERVICIOS DE AGUA POTABLE Y ALCANTARILLADO PARA LA JUNTA ADMINISTRADORA DE SERVICIO Y

SANEAMIENTO (JASS) DEL CENTRO POBLADO LURINCHINCHA EN EL DEPARTAMENTO DE ICA DURANTE EL AÑO 2026

Trabajo de aplicación profesional para obtener el título de Desarrollo de Sistemas de Información

JOSEPH MOISES IPANAQUE HUARI (0009-0007-3687-8674)

Lima – Perú 2026

A mis padres, hermanos y tíos, en especial a mi

madrina Piedad y a mi tío Cesar, por el constante apoyo e impulso para mi crecimiento

profesional. Y a Loki, mi fiel amigo de cuatro patas, quien me acompaño en mis primeros años como profesional.

6 10

Índice General

Pág.

Resumen Ejecutivo 10 Introducción 11 Capítulo I: Análisis del Negocio 12 Descripción de la Organización 12 Antecedentes 13 Visión Organizacional 14 Misión Organizacional 14 Organigrama y Funciones de los Principales Roles 15 Mapa de Procesos 17 Análisis FODA 18 Elicitación de Requisitos 19 Identificación de la Necesidad (Problema) 20 Análisis del Problema 20 Propuesta de Solución 21 Alternativas de Solución 22 Factibilidad del Proyecto 24 Factibilidad Técnica 24 Recursos Tecnológicos 25 Factibilidad Operativa 26 Factibilidad Económica 26 Análisis Costo-Beneficio 31 Acta de Constitución del Proyecto 33 Capítulo II: Planificación del Proyecto 35 Enfoque de Desarrollo y del Ciclo de Vida 35 Enfoque a aplicar 35 Justificación del enfoque a aplicar 36 Arquitectura de Software a Utilizar 36 Modelos, métodos y artefactos a aplicar 37 Planificación del Proyecto 39 Plan del Proyecto 39 Declaración de Alcance del Proyecto 40 Objetivos del Proyecto 41 Objetivos Generales 41 Objetivos Específicos 41 Beneficios del Proyecto 41 Diagrama de Gantt 42 Interesados 43 Supuestos 43 Factores Críticos de Éxito (FCE) 44 Riesgos 44 Comunicaciones 49 Capítulo III: Ejecución del Proyecto 50 Fase Inicio 50 Presentación del Lanzamiento del Proyecto (Kick Off) 50 Historias de Usuarios (HU) 53 Product Backlog 54 Fase Planificación 56 Historias de Usuarios Priorizados 56 Estimación de Tareas de las HU 58 Sprint Planning 64 Descomposición de Tareas de las HU 64 Fase Ejecución 73 Incremento del Sprint Backlog 73 Mantenimiento del Product Backlog 76 Dayli Scrum Meeting 77 Desarrollo del Spring Backlog 78 Fase de Revisión y Retrospectiva 79 Scrum Board 79 Validación de los Sprint (Sprint Review) 80 Burn Down Chart 81 Creación de la lista de impedimento de log 83 Retrospectivas (sprint y del Product Backlog) 84 Pruebas de Aceptación

85 Fase Transición y Cierre 90 Actas de Conformidad de los Entregables 90 Capítulo IV: Programación 91 Implementación de la Arquitectura de Software 91 Creación de la Base de Datos 92 Implementación de Librerías y Dependencias 93 Codificación del Back End 94 Codificación del Front End 98 Codificación de Consultas y Reportes 102 Codificación de Mantenedores (CRUD) 103 Codificación de Procesos Transaccionales 106 Capítulo V: Pruebas de Calidad de Software 107 Pruebas Unitarias 107 Pruebas Integrales 111 Conclusiones 115 Conclusión 1 115 Conclusión 2 115 Conclusión 3 115 Conclusión 4 116 Recomendaciones 117 Recomendación 1 117 Recomendación 2 117 Recomendación 3 117 Recomendación 4 118 Referencias Bibliográficas 119 Anexos 121 Anexo 1: Scrip de la Base de Datos 121 Anexo 2: Codificación Back-End 130

Índice de Tablas

Pág.

Tabla 1. Requisitos funcionales 19

Tabla 2. Requisitos no funcionales 19

Tabla 3. Factibilidad técnica para el desarrollo del proyecto - JAAS 24

Tabla 4. Recursos tecnológicos para el desarrollo del proyecto – JAAS 25

Tabla 5. Porcentaje de participación por roles en el proyecto 27

Tabla 6. Estimación de costos por roles (4 meses) 28

Tabla 7. Costos de servicios tecnológicos 29

Tabla 8. Costos de Hardware 29

Tabla 9. Costo de Software 30

Tabla 10. Precio Venta de Solución 30

Tabla 11. Ingreso promedio mensual sin sistema 31

Tabla 12. Ingreso promedio mensual con sistema 31

Tabla 13. Egresos e Ingresos mensuales sin sistema / con sistema (6 meses) 32

Tabla 14. Cálculo del beneficio mensual 32

Tabla 15. Análisis Costo/Beneficio 33

Tabla 16. Acta de Constitución del Proyecto 33

Tabla 17. Product Backlog 40

Tabla 18. Interesados del Proyecto 43

Tabla 19. Factor Crítico de Éxito 44

Tabla 20. Escala de probabilidad 44

Tabla 21. Escala de impacto 45

Tabla 22. Matriz de probabilidad e impacto 45

Tabla 23. Lista de Riegos 46

Tabla 24. Estrategia para mitigar los riegos 47

Tabla 25. Tabla de comunicaciones 49

Tabla 26. Historia de Usuario 53

Tabla 27. Product Backlog 55

Tabla 28. Historia de Usuario Priorizada 57

Tabla 29. Estimación de tareas de HU 60

Tabla 30. Sprint Planning 64

Tabla 31. Descomposición de tareas 64

Tabla 32. Plantilla Scrum 70

Tabla 33. Scrum Board: Sprint 1 79

Tabla 34. Scrum Board: Sprint 2 79

Tabla 35. Scrum Board: Sprint 3 80

Tabla 36. Scrum Board: Sprint 4 80

Tabla 37. Sprint Review de Sprint 1, 2, 3 y 4 80

Tabla 38. Impedimento de log 83

Tabla 39. Retrospective del Sprint 84

Tabla 40. Pruebas de Aceptación N° 1 85

Tabla 41. Pruebas de Aceptación N° 2 87

Tabla 42. Pruebas de Aceptación N° 3 88

Tabla 43. Pruebas de Aceptación N° 4 89

Tabla 44. Acta de conformidad de los entregables 90

Tabla 45. Librerías y Dependencias Utilizadas 93

Tabla 46. Respuestas HTTP 107

Tabla 47. Métodos HTTP 107

Tabla 48. Prueba de inicio de sesión 108

Tabla 49. Prueba de Listado de Consumidores 109

Tabla 50. Prueba de Creación de servicio especial 110

Índice de Figuras

Pág.

Figura 1. Ubicación de la JAAS del Centro Poblado Lurinchincha 14

Figura 2. Organigrama de la JASS 15

Figura 3. Mapa de Procesos 17

Figura 4. Diagrama actual (As-Is) del proceso “Administración Integral de Cliente y Servicios” de la JAAS 17

Figura 5. Análisis FODA de la JASS 18

Figura 6. Ciclo de vida Scrum 35

Figura 7. Artefactos Scrum 38

Figura 8. Plan del Proyecto 39

Figura 9. Diagrama de Gantt Planificado 42

Figura 10. Método de MOSCOW 56

Figura 11. Planning Poker 58

Figura 12. Aplicación de Planning Poker 59

Figura 13. Captura de Planning Poker 59

Figura 14. Incremento del Spring Backlog – Sprint 1 73

Figura 15. Incremento del Spring Backlog – Sprint 2 74

Figura 16. Incremento del Spring Backlog – Sprint 3 74

Figura 17. Incremento del Spring Backlog – Sprint 4 75

Figura 18. Mantenimiento del Product Backlog 76

Figura 19. Dayli Scrum Meeting 77

Figura 20. Desarrollo del Spring Backlog 78

Figura 21. Burn Down Chart – Sprint 1 81

Figura 22. Burn Down Chart – Sprint 2 82

Figura 23. Burn Down Chart – Sprint 3 82

Figura 24. Burn Down Chart – Sprint 4 83

Figura 25. Codificación de back-end de creación de roles 96

Figura 26. Codificación de back-end de creación de servicio especiales 97

Figura 27. Codificación de front-end de inicio de sesión 98

Figura 28. Codificación de front-end de creación del consumidor 99

Figura 29. Codificación de front-end de creación de roles 100

Figura 30. Codificación de front-end de creación de servicios especiales 101

Figura 31. Codificación de filtro de búsqueda de consumidores 102

Figura 32. Codificación de filtro de búsqueda de usuarios por dni 103

Figura 33. Codificación de creación de servicio especial 103

Figura 34. Codificación de actualizar servicio especial 104

Figura 35. Codificación de listar todos los servicios especiales 104

Figura 36. Codificación de eliminar servicio especial 105

Figura 37. Crear Consumidor 111

Figura 38. Modal de confirmación de creación de consumidor 112

Figura 39. Listado de consumidores 113

Figura 40. Pantalla de detalles de servicios asignado al consumidor 113

Figura 41. Pantalla de detalle del pago que deberá pagar el consumidor del periodo actual . 114 Figura 42. Clase Data-Initializer 130

Resumen Ejecutivo

La Junta Administradora de Servicio y Saneamiento (JASS) del Centro Poblado Lurinchincha, ubicada en el departamento de Ica, es la organización encargada de la gestión y administración del servicio de agua potable y alcantarillado en la comunidad. El proceso de negocio que se mejoró correspondió a la gestión administrativa, específicamente en las actividades de cobros, control de pagos, registro de deudas y elaboración de reportes financieros; que se realizaban de manera manual, ocasionando retrasos, errores y falta de información oportuna para la toma de decisiones. Esta situación evidenció la necesidad de modernizar y automatizar los procesos administrativos, con el fin de optimizar recursos, mejorar la eficiencia operativa y garantizar una administración más transparente del servicio. Se planteó como solución el desarrollo e implementación de un sistema web que permitiera centralizar los datos de los consumidores, registrar y controlar pagos y deudas, y generar reportes contables de forma automática y confiable. Con la puesta en marcha del sistema, la JASS logró reducir los tiempos de procesamiento, mejorar la precisión de la información, y fortalecer la gestión interna del servicio, contribuyendo al orden financiero y a la eficiencia en la administración del recurso hídrico. En conclusión, el proyecto permitió a la JASS del Centro Poblado Lurinchincha alcanzar una gestión más moderna, transparente y sostenible durante el año 2026.

Palabras clave: JASS; Centro Poblado Lurinchincha; agua potable; alcantarillado; gestión administrativa; cobros; pagos; deudas; reportes financieros; sistema web; automatización; eficiencia operativa; administración transparente; recurso hídrico; sostenibilidad.

Introducción

El presente documento expone el proceso de desarrollo e implementación de un sistema web orientado a optimizar la gestión de cobros por los servicios de agua potable y alcantarillado. Para llevar a cabo este proyecto, se adoptó un enfoque de gestión adaptativo basado en metodologías Ágiles, lo que permitió una ejecución iterativa y flexible. La solución técnica se construyó utilizando Spring Boot para el desarrollo del backend, Angular para el frontend, PostgreSQL versión 15 como sistema gestor de base de datos, y Java versión 17 como lenguaje principal de programación.

En el Capítulo I se desarrolla el análisis del negocio, presentando la descripción de la institución, su estructura organizacional, misión, visión y los procesos administrativos involucrados. Además, se identificó el problema, se realizó el análisis FODA y se propuso una solución evaluando su factibilidad técnica, operativa y económica.

El Capítulo II aborda la planificación del proyecto, detallando el enfoque metodológico, los objetivos, el alcance y los beneficios esperados. También se definieron el cronograma de actividades, los riesgos potenciales y los factores críticos de éxito que guiaron la ejecución ordenada del proyecto.

En el Capítulo III se describe la ejecución del proyecto, documentando las fases de desarrollo, seguimiento, control y validación. El Capítulo IV presenta la programación del sistema, explicando la implementación de la arquitectura de software y los módulos desarrollados. Finalmente, el Capítulo V expone las pruebas de calidad, las conclusiones y recomendaciones que demostraron la mejora alcanzada en la gestión de cobros de la JASS del Centro Poblado Lurinchincha durante el año 2026.

Capítulo I:

Análisis del Negocio

Descripción de la Organización

La Junta Administradora de Servicio y Saneamiento (JASS) del Centro Poblado Lurinchincha, ubicada en el distrito de Chincha Baja, provincia de Chincha, departamento de Ica, es una organización comunal responsable de administrar y supervisar la prestación de los servicios de agua potable y alcantarillado en su jurisdicción. Su labor se orienta a garantizar el acceso continuo y seguro al recurso hídrico, así como al mantenimiento del pozo, motor, redes de distribución, tuberías y demás infraestructuras que conforman el sistema de abastecimiento.

Además, la JASS tiene a su cargo la gestión de cobros y el control de pagos por los servicios brindados, con el objetivo de asegurar la sostenibilidad económica y operativa del sistema. La institución promueve una administración transparente y participativa, en la que la comunidad desempeña un rol activo en la toma de decisiones y en la conservación de los recursos.

A través de su gestión, la JASS del Centro Poblado Lurinchincha contribuye al bienestar y desarrollo de la población local, velando por la eficiencia en la distribución del agua y el adecuado funcionamiento del sistema de saneamiento básico, pilares fundamentales para la salud pública y la calidad de vida de los habitantes.

Antecedentes

La Junta Administradora de Servicio y Saneamiento (JASS) del Centro Poblado Lurinchincha se fundó en el año 2006, con el propósito de organizar y garantizar el acceso al agua potable y al saneamiento básico para las familias del centro poblado. Su creación respondió a la necesidad de contar con una entidad comunal que gestione, mantenga y supervise la infraestructura de distribución del agua y alcantarillado.

En sus primeros años, la JASS inició con un pequeño equipo de trabajo conformado por vecinos del lugar, quienes asumieron voluntariamente los cargos de presidente, secretario, tesorero y vocales. Gracias a su esfuerzo y compromiso, la organización logró consolidarse como una institución reconocida dentro de la comunidad.

Con el paso del tiempo, la JASS ha fortalecido su estructura administrativa y operativa, promoviendo la participación de los pobladores en la toma de decisiones y la sostenibilidad del servicio. Actualmente, continúa desempeñando un papel fundamental en la mejora de la calidad de vida de los habitantes, garantizando un servicio eficiente, transparente y orientado al bienestar común.

Figura 1:

Ubicación de la JAAS del Centro Poblado Lurinchincha

Visión Organizacional

La JASS del Centro Poblado Lurinchincha busca ser una organización modelo en la gestión sostenible del agua y saneamiento, destacando por su eficiencia, transparencia y compromiso con el bienestar de la comunidad.

Misión Organizacional

La JASS del Centro Poblado Lurinchincha administra y garantiza el acceso equitativo a los servicios de agua potable y saneamiento, mediante una gestión responsable, transparente y sostenible, comprometida con la salud y el bienestar de la comunidad.

Organigrama y Funciones de los Principales Roles

El organigrama de la Junta Administradora de Servicios y Saneamiento (JASS) del Centro Poblado Lurinchincha se basa en una estructura jerárquica de tres niveles, que permite una distribución clara de responsabilidades y una adecuada coordinación entre las áreas administrativas y operativas. En la parte superior se encuentra el presidente, como máxima autoridad y representante legal de la institución. En el segundo nivel se ubican el secretario y el tesorero, encargados de la gestión administrativa y financiera, respectivamente. Finalmente, en el tercer nivel se encuentran los vocales y el encargado de mantenimiento, quienes apoyan en la supervisión de actividades y en la operación técnica del sistema de agua y saneamiento.

Figura 2

Organigrama de la JASS

Presidente
Secretario
Tesorero
Vocal 1
Vocal 2
Encargado de Mantenimiento

Funciones de los Principales Roles Roles y Funciones

Presidente

Representa legalmente a la JASS, dirige las reuniones, supervisa las actividades administrativas y vela por el cumplimiento de los objetivos institucionales.

Secretario

Registra los acuerdos, redacta actas, gestiona la documentación institucional y mantiene actualizados los registros de la organización.

Tesorero

Administra los recursos económicos, controla los ingresos y egresos, elabora informes financieros y garantiza la transparencia contable.

Vocales (1 y 2)

Apoyan en la ejecución de las actividades, supervisan tareas operativas y reemplazan a los titulares cuando corresponde.

Encargado de Mantenimiento

Realiza la operación y mantenimiento del pozo, redes de distribución, válvulas y demás infraestructura del sistema de agua y saneamiento.

Mapa de Procesos

Figura 3

Mapa de Procesos

Figura 4

Diagrama actual (As-Is) del proceso "Administración Integral de Cliente y Servicios" de la JAAS

Análisis FODA

Figura 5

Análisis FODA de la JASS.

FORTALEZAS

· Personal comprometido con el bienestar de la comunidad.

· Existencia de registros históricos de usuarios y pagos, aunque sean manuales.

· Comunicación directa entre la junta directiva y los consumidores.

OPORTUNIDADES

· Implementación de herramientas tecnológicas accesibles y económicas (sistemas web).

· Crecimiento de la comunidad y aumento de usuarios del servicio.

· Integración futura con otros sistemas (pasarela de pagos, etc.).

FODA

DEBILIDADES

· Registro manual de pagos y deudas que genera errores y pérdidas de información.

· Dificultad para generar reportes financieros

actualizados y confiables.

AMENAZAS

· Incremento en costos operativos o falta de recursos económicos para mantenimiento del sistema.

Nota: Adaptado de Raeburn, A. (2025). Análisis FODA: qué es y cómo usarlo (con ejemplos).

Asana. https://asana.com/es/resources/swot-analysis

Elicitación de Requisitos

Tabla 1

Requisitos funcionales.

REQUERIMIENTOS FUNCIONALES (RF)	
RF	**DETALLE**
RF01	El sistema debe permitir registrar, actualizar y eliminar consumidores.
RF02	El sistema debe permitir registrar pagos mensuales y verificar deudas anteriores. RF03 El sistema debe calcular automáticamente la deuda total del consumidor.

RF04 El sistema debe permitir registrar pagos parciales y mantener saldo pendiente.

RF05 El sistema debe generar recibos de pago.

RF06 El sistema debe permitir consultar el historial de pagos de cada consumidor. RF07 El sistema debe permitir generar reportes financieros por mes, año o rango de fechas.

RF08 El sistema debe permitir gestionar roles de usuario (administrador, tesorero, secretario, consumidor).

RF09 El sistema debe permitir al consumidor ingresar con usuario y contraseña para consultar su deuda.

RF10 El sistema debe registrar los servicios especiales (corte, reconexión, mantenimiento) y su impacto en la deuda.

Tabla 2

Requisitos no funcionales.

REQUERIMIENTOS NO FUNCIONALES (RNF)

RNF **DETALLE**

RNF01 El sistema debe tener una interfaz web de tipo responsivo (accesible desde computadoras y dispositivos móviles).

RNF02 La información debe almacenarse en una base de datos segura y respaldada periódicamente. RNF03 El sistema debe permitir autenticación mediante credenciales seguras (JWT o similar).

RNF04 El sistema debe garantizar tiempos de respuesta menores a 3 segundos por operación.

RNF05 El sistema debe generar reportes exportables (PDF o Excel).

RNF06 El sistema debe registrar auditorías básicas (fecha, usuario, acción).

RNF07 El sistema debe ser modular y escalable para agregar nuevos servicios en el futuro.

Identificación de la Necesidad (Problema)

En el Centro Poblado Lurinchincha, la Junta Administradora de Servicios de Saneamiento (JASS) es la entidad encargada de gestionar los servicios de agua potable y alcantarillado.

Actualmente, la administración de los cobros, pagos y control de usuarios se realiza de forma manual, utilizando cuadernos. Este método tradicional genera múltiples dificultades, como la pérdida de información, errores en los registros, falta de trazabilidad en los pagos y demoras en la elaboración de reportes financieros.

La ausencia de un sistema automatizado ha limitado la eficiencia de los procesos administrativos y ha dificultado la

transparencia en la gestión económica. Además, los usuarios no cuentan con un medio rápido ni confiable para consultar su estado de deuda o sus pagos realizados, lo que ocasiona retrasos y reclamos frecuentes. Estas deficiencias impactan directamente en la sostenibilidad financiera de la JASS y en la calidad del servicio brindado a la comunidad.

Análisis del Problema

El método de los cinco porqués permite descubrir la causa raíz de un problema mediante una secuencia lógica de preguntas *(Ishikawa, 1985)* .

Problema principal:

La gestión manual de registro, cobros y pagos del servicio de agua potable y alcantarillado en la JASS del Centro Poblado Lurinchincha ocasiona demoras, errores y dificultades para generar reportes confiables.

1. ¿Por qué existen demoras y errores en el registro de pagos?

Porque los cobros y registros se realizan manualmente, utilizando cuadernos o planillas en papel.

2. ¿Por qué se realiza el registro de forma manual?

Porque no se cuenta con un sistema informático que permita registrar y administrar la información de manera automatizada.

3. ¿Por qué no se cuenta con un sistema informático?

Porque la organización carece de herramientas tecnológicas y no ha implementado un proceso de modernización digital.

4. ¿Por qué no se ha implementado la modernización digital?

Porque el personal administrativo no dispone de conocimientos técnicos ni de asesoramiento para aplicar soluciones tecnológicas de gestión.

5. ¿Por qué no se dispone de conocimientos técnicos ni asesoramiento?

Porque la JASS funciona de manera comunal y autogestionada, dependiendo de recursos limitados y sin apoyo institucional permanente.

Propuesta de Solución

Para resolver los problemas generados por el manejo manual de los cobros, pagos y control de usuarios, se propone el desarrollo e implementación de un sistema web. El sistema permitirá registrar consumidores, generar cobros mensuales, controlar pagos, deudas, y emitir reportes automáticos de manera rápida y segura. Con ello, se busca reducir errores, agilizar los procesos administrativos y mejorar la transparencia en la gestión de los recursos.

Alternativas de Solución

Alternativa 1: Uso de hojas de cálculo en Microsoft Excel o Google Sheets

En esta alternativa se propone mantener el registro de pagos, cobros, usuarios y reportes financieros mediante hojas de cálculo, aprovechando herramientas como Microsoft Excel o Google Sheets. Esta opción tiene la ventaja de ser de bajo costo y rápida implementación, pues solo requiere una computadora y conocimientos básicos del programa. Sin embargo, presenta limitaciones importantes, como la falta de seguridad, riesgo de pérdida o duplicidad de información, y una alta dependencia del manejo manual. Además, al aumentar la cantidad de usuarios, el control se vuelve más complejo y propenso a errores. El costo estimado es mínimo (S/. 0 a S/. 500), considerando únicamente los recursos informáticos y tiempo de capacitación, con una implementación casi inmediata, aunque su sostenibilidad a largo plazo es limitada.

Alternativa 2: Implementación de una plataforma autoadministrada (no especializada)

Esta alternativa consiste en emplear una plataforma genérica de gestión, como sistemas de facturación o control contable disponibles en línea, para registrar pagos y emitir recibos. Su principal ventaja es que permite una implementación más rápida que un desarrollo propio y ofrece ciertas funciones automatizadas. No obstante, sus desventajas radican en que no se adapta completamente a las necesidades específicas de las JASS, como el control de morosidad, conexión y reconexión del servicio o reportes personalizados por consumidor. Además, puede implicar costos de suscripción o licencias periódicas. El costo aproximado sería de S/. 2 000 a S/.

5 000 según el proveedor y nivel de personalización, con un tiempo de implementación estimado de una a tres semanas y restricciones relacionadas con la dependencia de soporte externo y la limitada capacidad de adaptación.

Factibilidad del Proyecto

Factibilidad Técnica

Evalúa si el proyecto puede desarrollarse con los conocimientos, herramientas y condiciones operativas disponibles. (Raeburn 2025)

Tabla 3

Factibilidad técnica para el desarrollo del proyecto - JAAS

Aspecto Evaluado	Descripción
Competencia del recurso humano	Los integrantes de la organización cuentan con conocimientos básicos para el manejo de herramientas digitales, lo que les permite aprender y operar adecuadamente el sistema con una capacitación mínima.
Compatibilidad de herramientas	Actualmente la organización carece de equipos informáticos e internet. Para garantizar el uso del sistema será necesaria la adquisición de una computadora y acceso a internet, lo cual es viable técnica y logísticamente dentro de la organización.
Viabilidad operativa	Una vez adquiridos los recursos tecnológicos básicos (equipo + internet), la organización podrá operar el sistema de manera regular. El proceso operativo se ajusta a las actividades actuales de gestión administrativa de la JASS.
Escalabilidad potencial	La organización tiene la posibilidad de ampliar sus recursos tecnológicos en caso de requerirse (más equipos, mejor internet), lo cual permitiría atender a más usuarios o mayor volumen de operaciones sin alterar su funcionamiento interno.
Sostenibilidad técnica	Mantener el funcionamiento del sistema dentro de la organización es viable, siempre que se garantice el mantenimiento básico del equipo y el acceso a internet. Para aspectos técnicos avanzados se puede recurrir a soporte externo.

El proyecto es técnicamente viable, ya que se dispone de las herramientas necesarias para su desarrollo, las cuales son de uso libre y ampliamente documentadas. Además, el sistema será implementado en un entorno web, lo que permitirá su acceso desde cualquier dispositivo con conexión a Internet, sin requerir instalaciones adicionales.

El mantenimiento técnico y la gestión de datos podrán ser supervisados por un responsable designado, sin necesidad de contratar personal adicional.

Recursos Tecnológicos

Lista los elementos humanos, de hardware y software necesarios para ejecutar el proyecto. (González 2023)

Tabla 4

Recursos tecnológicos para el desarrollo del proyecto – JAAS

Tipo de Recurso	Nombre del Recurso	Descripción / Función	Cantidad

Hardware	Laptop / PC de desarrollo	Equipo con procesador Intel Core i5 o superior, 16 GB de RAM y SSD de 512 GB. Utilizado para el desarrollo y pruebas del sistema.	1
Hardware	Servidor web	Espacio en servidor remoto (puede ser AWS, Google Cloud o similar) para desplegar el sistema y permitir acceso vía internet.	1
Software	IntelliJ IDEA	Entorno de desarrollo integrado (IDE) utilizado para la programación del backend con Spring Boot.	1
Software	Visual Studio Code	Entorno de desarrollo para la creación del frontend en Angular.	1
Software	PostgreSQL	Sistema de gestión de base de datos relacional (open source) utilizado para almacenar y gestionar la información.	1
Software	Git / GitHub	Herramienta de control de versiones y repositorio remoto para el trabajo colaborativo y respaldo del código fuente.	1
Software	Postman	Herramienta para pruebas y validación de los endpoints del backend.	1
Software	Windows 11	Sistema operativo del entorno de desarrollo.	1
Software	Navegadores Web	Herramientas para probar la interfaz web del sistema.	1

La lista anterior describe las necesidades de infraestructura tecnológica que va a requerir el proyecto y que la organización no cuenta.

Factibilidad Operativa

La factibilidad operativa evalúa la capacidad del Centro Poblado Lurinchincha para incorporar y utilizar de manera efectiva el sistema web propuesto dentro de sus actividades administrativas.

Recursos Humanos. El personal del Centro Poblado Lurinchincha, conformado por el presidente, tesorero y secretario, serán los principales usuarios del sistema. Estos cuentan con conocimientos básicos en el manejo de computadoras e Internet, por lo que solo requerirán una capacitación breve sobre el uso del sistema.

Aceptación del Sistema. El sistema responde a una necesidad real del área administrativa, por lo que se espera una alta aceptación entre los usuarios. La automatización de los cobros y reportes mejorará la eficiencia operativa y reducirá los conflictos relacionados con errores de registro o cálculos manuales.

Factibilidad Económica

La factibilidad económica analiza la viabilidad del proyecto desde el punto de vista financiero, considerando los recursos requeridos para su desarrollo, implementación y mantenimiento (Shopify, 2025). El sistema web propuesto representa una inversión accesible para la JASS, ya que emplea herramientas de código abierto y no requiere licencias costosas.

Tabla 5

Porcentaje de participación por roles en el proyecto

Rol	Responsabilidades principales	Mes 1 (%)	Mes 2 (%)	Mes 3 (%)	Mes 4 (%)	Total (%)
Product Owner	Definir y priorizar requerimientos, gestionar backlog, validar entregables con el cliente.	40%	30%	20%	10%	**100%**
Scrum Master	Facilitar eventos Scrum, eliminar impedimentos, asegurar buenas prácticas.	25%	25%	25%	25%	**100%**
Development Team	Diseño UI/UX, análisis, desarrollo backend y frontend, pruebas, documentación técnica.	35%	45%	55%	65%	**200%**
Total		**100%**	**100%**	**100%**	**100%**	**400%**

Tabla 6

Estimación de costos por roles (4 meses)

Rol	Costo / Hora (S/.)	Mes 1 (S/.)	Mes 2 (S/.)	Mes 3 (S/.)	Mes 4 (S/.)	Total (S/.)				
		Horas	Costo (S/.)	Horas	Costo (S/.)	Horas	Costo (S/.)	Horas	Costo (S/.)	
Product Owner (50% Líder)	40	64	2,560	48	1,920	32	1,280	16	640	**2,720**
Scrum Master (50% Líder)	40	40	1,600	40	1,600	40	1,600	40	1,600	**2,640**
Development Team	30	56	1,680	72	2,160	88	2,640	104	3,120	**2,850**
TOTAL (S/.)			5,840		5,680		5,520		5,360	**22,400**

Tabla 7

Costos de servicios tecnológicos

Servicio	Descripción	Costo mensual (S/.)	X 4 Meses (S/)
Servidor cloud (VM + DB pequeña)	Instancia para la aplicación y base de datos (VPS o instancia cloud pequeña).	150	600
Backup / Almacenamiento	Almacenamiento para copias automáticas y restauración (S3 u opción similar).	30	120

Dominio + SSL	Dominio (.com/.pe) prorrateado y certificado SSL (Let's Encrypt gratuito; si se usa pago, aumentaría).	5	20
Monitoring / Logs / Alertas	Servicio básico de monitoreo, alertas y gestión de logs.	30	120
TOTAL		S/ 215 / mes	**S/ 860.0**

Tabla 8

Costos de Hardware

Equipo / Dispositivo	**Especificaciones Técnicas**	**Cantidad**	**Costo Unitario (S/.)**	**Costo Total (S/.)**
Laptop para desarrollo	Intel Core i5 o superior, 16 GB RAM, 512 GB SSD	1	3,500	3,500
Laptop para pruebas funcionales / QA	Intel Core i5, 8 GB RAM, 256 GB SSD	1	2,800	2,800
Router / Conexión de red estable	Velocidad mínima 100 Mbps	1	250	250
TOTAL				**S/6,900**

Tabla 9

Costo de Software

Software / Herramienta	**Uso principal**	**Licencia / Tipo**	**Costo (S/.)**	**Periodicidad**
IntelliJ IDEA Community Edition	Desarrollo Backend (Spring Boot)	Gratuito / Open Source	0	-
Visual Studio Code	Desarrollo Frontend (Angular)	Gratuito / Open Source	0	-
PostgreSQL	Base de datos relacional	Gratuito / Open Source	0	-
Git / GitHub	Control de versiones	Gratuito / Plan básico	0	-
Microsoft Office (Word, Excel)	Documentación y reportes	Licencia anual	400	Anual
SO Windows 11 Pro	Plataforma de trabajo	Licencia original	600	Única

Antivirus	Protección del equipo	Licencia anual	100	Anual
TOTAL			S/1,100	

Tabla 10

Precio Venta de Solución

Categoría	**Costo Total (S/.)**
Roles	22,400.0
Servicios Tecnológicos	860.0
Hardware	6,900.0
Software	1,100.0
subtotal	31,290.0
Margen 40%	12,516.0
Costo Venta	43,806.0
IGV (18%)	7,885.08
Precio Venta	**51,691.08**

Análisis Costo-Beneficio

Tabla 11

Ingreso promedio mensual sin sistema

Ingresos promedio mensual sin sistema

Concepto Descripción Monto (S/.)

Recaudación mensual promedio por consumo de agua

Pagos por reconexión del servicio Aportes extraordinarios o mantenimiento
Aportes ocasionales destinados a reparaciones o mejoras.

150

Pagos realizados manualmente por los usuarios del 2,500 servicio.

Ingresos por penalidades y reconexiones realizadas. 350

Total, promedio mensual sin sistema

3,000

Tabla 12

Ingreso promedio mensual con sistema

Ingresos promedio mensual con sistema

Concepto Descripción Monto (S/.)

Recaudación mensual promedio por consumo de agua
Incremento del 20% por mejora en el control de pagos y notificaciones.
3,000

Pagos por reconexión del servicio Reducción de morosidad y mejor gestión de

reconexiones.

400

Aportes extraordinarios o mantenimiento
Mayor transparencia y participación comunitaria.
200

Total, promedio mensual con

sistema

3,600

Tabla 13

Egresos e Ingresos mensuales sin sistema / con sistema (6 meses)

Egresos e Ingresos mensuales sin sistema / con sistema (6 meses)						
Ingresos Egresos Utilidad Ingresos	**Egresos**	**Utilidad**				
Mes sin sistema sin sistema sin sistema con	**con**	**con**				
(S/.) (S/.) (S/.) sistema	**sistema**	**sistema**				
(S/.)	**(S/.)**	**(S/.)**				
Mes 1	3,000	750	2,250	3,600	580	3,020
Mes 2	3,000	750	2,250	3,600	580	3,020
Mes 3	3,000	750	2,250	3,600	580	3,020
Mes 4	3,000	750	2,250	3,600	580	3,020
Mes 5	3,000	750	2,250	3,600	580	3,020
Mes 6	3,000	750	2,250	3,600	580	3,020
Totales (6 meses)	**18,000**	**4,500**	**13,500**	**21,600**	**3,480**	**18,120**

Nota. Ingresos e Egresos con valores promedios

Tabla 14

Cálculo del beneficio mensual

Cálculo de beneficio mensual

Descripción Fórmula Resultado (S/.)

Incremento de ingresos mensuales
Ingreso con sistema – Ingreso sin sistema
3,600 – 3,000 = **600**

Ahorro en costos mensuales Costo sin sistema – Costo con

sistema

750 – 580 = **170**

Beneficio total mensual
Incremento de ingresos + Ahorro en costos
600 + 170 = **770**

Tabla 15

Análisis Costo/Beneficio

Beneficio / Costo =	**770 / 580**	
Beneficio / Costo =	**1.33**	

Un valor B/C = 1.33 indica que, por cada sol invertido en la operación del sistema, se obtiene un beneficio de S/ 1.33, lo cual demuestra que el proyecto es rentable y viable económicamente.

Acta de Constitución del Proyecto

Tabla 16

Acta de Constitución del Proyecto

Nombre del Proyecto	**Siglas**
Sistema Web de Gestión de Cobros, Pagos y Reportes para la JASS del Centro Poblado Lurinchincha	**SISWEB-JASS**
Descripción del Proyecto	
Desarrollo e Implementación De Un Sistema Web Para Optimizar La Gestión De Cobros Y Reportes Por Servicios De Agua Potable Y Alcantarillado Para La Junta Administradora De Servicio Y Saneamiento (JASS) Del Centro Poblado Lurinchincha En El Departamento De Ica Durante El Año 2026	
Definición del Producto del Proyecto	
El producto será una aplicación web desarrollada con **Spring Boot (backend)** , **Angular (frontend)** y **PostgreSQL (base de datos)** , accesible desde cualquier dispositivo con conexión a internet. Permitirá al equipo administrativo gestionar los pagos, emitir reportes y mantener un registro actualizado de los consumidores.	
Objetivo del Proyecto	
Diseñar e implementar un sistema web que optimice la gestión administrativa y financiera de la JASS, reduciendo errores manuales y mejorando la transparencia en el control de pagos y reportes.	
Restricciones	
- Cumplir con los plazos establecidos.	
- Mantener estándares de seguridad en la gestión de datos.	

- Adaptar el sistema al entorno tecnológico disponible.

Justificación del Proyecto	
Actualmente, los registros y cobros se realizan de forma manual, generando errores y demoras en la administración. El sistema permitirá centralizar la información, agilizar procesos y fortalecer la transparencia financiera de la JASS.	
Designación del Equipo del Proyecto	
Cliente: JASS Centro Poblado Lurinchincha	
Equipo de Desarrollo: Joseph Moises Ipanaque Huari	
Cronograma de Hitos del Proyecto	
- Fase de Inicio: 01/11/2025 – 15/11/2025	
- Fase de Planificación: 16/11/2025 – 15/12/2025	
- Fase de Implementación: 16/12/2025 – 31/01/2026	
- Fase de Revisión y Retrospectiva: 01/02/2026 – 10/02/2026	
- Fase de Lanzamiento: 11/02/2026 – 20/02/2026	
Organizaciones o Grupos que Intervienen en el Proyecto	
Nombre de la Organización	**Rol Dentro del Proyecto**
JASS Centro Poblado Lurinchincha	Entidad solicitante y beneficiaria
Equipo de Desarrollo	Diseño, programación, pruebas e implementación del sistema
Principales Riesgos del Proyecto	
- Fallas técnicas o limitaciones de conectividad.	
- Retrasos en la entrega de información por parte del cliente.	
- Disponibilidad limitada del personal de la JASS.	
Principales Oportunidades del Proyecto	
- Mejora de la transparencia y control financiero.	
- Fortalecimiento de la gestión comunitaria.	
- Facilidad de acceso a información en tiempo real.	
Presupuesto Preliminar del Proyecto	
Presupuesto en S/ **51,691.08**	
Presupuesto en $ **14,358.63**	
Patrocinador que Autoriza el Proyecto	
Nombre:	Presidente de la JASS del Centro Poblado Lurinchincha
Empresa/Entidad:	JASS del Centro Poblado Lurinchincha
Fecha:	25/10/2025
Firma del Patrocinador Presidente de la JASS	**Firma del Representante del**

Capítulo II:

Planificación del Proyecto

Enfoque de Desarrollo y del Ciclo de Vida

Enfoque a aplicar

Para la ejecución del proyecto se empleará el marco de trabajo ágil Scrum, debido a su naturaleza iterativa e incremental, orientada a la entrega continua de valor y a la adaptación frente a cambios. Este enfoque permite priorizar funcionalidades y asegurar que los incrementos entregados respondan a las necesidades reales de los usuarios.

Según SCRUMstudy™ (2022), Scrum es un enfoque adaptable, flexible y eficaz para gestionar proyectos en entornos dinámicos y con requerimientos cambiantes.

Figura 6

Ciclo de vida Scrum

Nota: Tomado de ¿Qué es y cómo funciona la metodología scrum? (29 abril 2020) https://www.arrobasolutions.com/metodologia-scrum/

Justificación del enfoque a aplicar

El marco de trabajo Scrum se ha seleccionado debido a su adaptabilidad, colaboración continua, entrega incremental de valor y mejora sostenida (SCRUMstudy™, 2022). Este enfoque no solo permite cumplir con los requisitos técnicos y operativos del comité administrador, sino también adaptarse dinámicamente a las expectativas de los usuarios finales y las condiciones en constante evolución del sistema de gestión de servicios de agua potable. Además, protocolos de práctica real muestran que la flexibilidad inherente a Scrum favorece la adaptación efectiva a cambios y promueve la comunicación constante entre los involucrados, lo que contribuye a alcanzar mejores resultados (Masood, Hoda, & Blincoe, 2021).

Arquitectura de Software a Utilizar

El sistema web se desarrollará bajo una arquitectura Cliente–Servidor, donde el frontend gestiona la presentación y el backend procesa la lógica de negocio y el acceso a datos. Este enfoque distribuye responsabilidades, facilita la escalabilidad y mejora el mantenimiento del sistema (Sommerville, 2016; Pressman & Maxim, 2020).

Frontend – Angular

El frontend utilizará Angular, empleando una organización modular basada en componentes y servicios, lo que facilita la reutilización y mantenibilidad de la interfaz. Angular integra conceptos derivados del patrón MVC, separando presentación y lógica (Google Developers, 2023).

Backend – Spring Boot con Clean Architecture

El backend se implementará con Spring Boot, siguiendo principios de Clean Architecture, la cual promueve bajo acoplamiento y alta cohesión mediante capas como dominio, casos de uso, adaptadores e infraestructura (Martin, 2018). La comunicación con el frontend se gestionará mediante servicios REST (Richardson, 2020).

Base de Datos – PostgreSQL

La información será gestionada por PostgreSQL, un motor relacional robusto, seguro y extensible, adecuado para aplicaciones transaccionales (Momjian, 2022).

Modelos, métodos y artefactos a aplicar

Para el desarrollo del sistema web, se aplicará un modelo incremental basado en la metodología ágil Scrum, lo que permitirá incorporar mejoras de forma continua durante la construcción del software (SCRUMstudy™, 2022).

El proyecto se llevará a cabo mediante iteraciones llamadas Sprints, donde se planificarán, desarrollarán y evaluarán incrementos funcionales del sistema. Al inicio se realizará un Sprint Planning, en la que el Product Owner y el equipo de desarrollo seleccionarán las historias de usuario a trabajar. Durante el Sprint se realizarán reuniones diarias Daily Scrum, enfocadas en revisar el avance, identificar impedimentos y replanificar tareas. Al finalizar cada Sprint, se efectuará un Sprint Review, en la que se validará el incremento y se recogerá retroalimentación para ajustar el Product Backlog según sea necesario. Finalmente, un Sprint Retrospective permitirá identificar oportunidades de mejora en el proceso, fortaleciendo la entrega continua de valor al usuario (SCRUMstudy™, 2022).

Artefactos a aplicar

Los principales artefactos utilizados en el desarrollo del proyecto serán:

Product Backlog: Lista priorizada y dinámica de los requerimientos funcionales y no funcionales del producto, gestionada por el Product Owner (SCRUMstudy™, 2017, p. 179).

Sprint Backlog: Conjunto de elementos seleccionados del Product Backlog para desarrollar durante el Sprint, junto con un plan de ejecución para alcanzar el objetivo propuesto (SCRUMstudy™, 2017, p. 223).

Incremento: Suma de todos los elementos completados durante un Sprint, que representan funcionalidad terminada y potencialmente utilizable (García, s. f.).

Estos artefactos proporcionan transparencia, orden y trazabilidad del trabajo, facilitando el control del progreso del proyecto.

Figura 7

Artefactos Scrum

Nota: Tomado de Artefactos Scrum. https://blog.buhoos.com/artefactos-de-scrum/

Planificación del Proyecto

Plan del Proyecto

Figura 8

Plan del Proyecto

Nota: Adaptado Guía SBOK® Tercera Edición: Fundamentos de Scrum, por SCRUMstudy, 2017, pág. 38.

Tabla 17

Product Backlog

Épica	**HU ID**	**Historia de Usuario**
	HU01	Registrar consumidores
Gestión de Consumidores	HU02	Actualizar consumidores
	HU03	Eliminar consumidores
	HU04	Registrar pagos mensuales
	HU05	Verificar deudas anteriores
	HU06	Calcular automáticamente la deuda total
Gestión de Pagos y Deudas	HU07	Registrar pagos parciales
	HU08	Consultar saldo pendiente
	HU09	Generar recibos de pago
	HU10	Consultar historial de pagos (tesorero)
Portal del Consumidor	HU11	Consultar historial de pagos (consumidor)
	HU12	Generar reportes mensuales
Reportes Financieros	HU13	Generar reportes anuales o por rango de fechas

Gestión de Roles y Usuarios	HU14	Gestionar roles del sistema
	HU15	Asignar roles a los usuarios
	HU16	Iniciar sesión con usuario y contraseña
Autenticación y Seguridad	HU17	Consultar deuda desde el portal del consumidor
Servicios Especiales	HU18	Registrar servicios especiales (corte, reconexión, mantenimiento)
	HU19	Ver historial de servicios especiales

Declaración de Alcance del Proyecto

El proyecto consiste en desarrollar un sistema web para la JASS que permita gestionar consumidores y sus pagos mensuales, calcular deudas y registrar servicios adicionales como corte, reconexión y mantenimiento. El sistema permitirá administrar consumidores, registrar pagos completos o parciales, generar recibos, consultar historial y emitir reportes financieros, además de gestionar roles de usuario y acceso de los consumidores para consultar su

información. La solución contará con interfaz web, base de datos segura, autenticación mediante credenciales seguras, generación de reportes exportables, auditoría básica y arquitectura modular y escalable. No incluye integraciones con sistemas externos, pagos en línea, lectura automática de medidores ni aplicación móvil.

Objetivos del Proyecto

Objetivos Generales

Desarrollar e implementar un sistema web que permita gestionar consumidores, registrar pagos mensuales, calcular deudas, administrar cargos adicionales (corte, reconexión, mantenimiento) y generar reportes financieros, garantizando seguridad, accesibilidad y facilidad de uso para la JASS.

Objetivos Específicos

· Realizar el análisis del negocio.

· Diseñar la solución tecnológica, incluyendo arquitectura Frontend, Backend y Base de Datos.

· Desarrollar el sistema web, implementando las funcionalidades principales definidas en el alcance.

· Realizar las pruebas de calidad del software, incluyendo pruebas unitarias y de integración.

· Implementar el despliegue y puesta en producción del sistema.

Beneficios del Proyecto

El proyecto permitirá agilizar significativamente el proceso de cobro de los servicios, reduciendo errores humanos y mejorando la eficiencia operativa. Además, al centralizar la información en una base de datos segura, se evitará la pérdida o deterioro de registros físicos,

garantizando mayor seguridad, disponibilidad y control sobre los datos, así como disminuyendo costos asociados a la gestión manual.

Diagrama de Gantt

Figura 9

Diagrama de Gantt Planificado

Actividades	**Duración del Proyecto**															
	Noviembre	**Diciembre**	**Enero**	**Febrero**												
	Semana	**Semana**	**Semana**	**Semana**												
	1	**2**	**3**	**4**	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**	**1**	**2**	**3**	**4**
Inicio																
Analisís del Problema																
Objetivo y Alcance del Proyecto																
Visión del Proyecto																
Crear Historias de Usuario																
Crear Product Backlog																
Planificación																
Enfoque de Trabajo de Utilizar																
Definir la Arquitectura de Software																
Historias de Usuario																
Criterios de Aceptación																
Product Backlog Priorizado y Estimado																
Descomposición de Tareas																
Implementación																
Sprint Planning 1																

Sprint 1															
Sprint Planning 2															
Sprint 2															
Sprint Planning 3															
Sprint 3															
Sprint Planning 4															
Sprint 4															
Revisión y Retrospectiva															
Sprint Review 1															
Sprint Retrospectiva 1															
Sprint Review 2															
Sprint Retrospectiva 2															
Sprint Review 3															
Sprint Retrospectiva 3															
Sprint Review 4															
Sprint Retrospectiva 4															
Lanzamiento															
Actas de Conformidad de los Entregables															
Capacitación de Usuario															
Pase a Producción															

Interesados

Tabla 18

Interesados del Proyecto

Interesados	Interés del Proyecto

Directiva de la JASS	Mejorar el control financiero, reducir errores humanos, agilizar cobros y tener información clara para la toma de decisiones.
Consumidores	Acceder de forma clara a su deuda, historial de pagos y mejorar la experiencia al realizar pagos.
Equipo de mantenimiento	Disponer de información oportuna sobre usuarios con deudas o cortes para gestionar reconexiones y mantenimiento.

En conclusión, los principales interesados del proyecto, la Directiva de la JASS, los consumidores y el equipo de mantenimiento, buscan mejorar la gestión operativa y la experiencia en el acceso al servicio. Por ello, es fundamental considerar sus necesidades y expectativas durante el desarrollo e implementación del sistema, garantizando transparencia en la administración, facilidad de acceso a la información y una atención eficiente para asegurar el éxito del proyecto.

Supuestos

· La Directiva de la JASS brindará información oportuna y completa sobre consumidores, tarifas y pagos históricos.

· Los usuarios contarán con acceso a internet para utilizar el sistema web.

· El equipo de desarrollo tendrá disponibilidad para reuniones de validación durante el proyecto.

· Los lineamientos de cobro y tarifas del servicio de agua se mantendrán estables durante la fase de desarrollo.

· La infraestructura tecnológica (servidor, hosting y base de datos) estará disponible para implementación.

· Los consumidores proporcionarán información verídica para su registro.

· El personal administrativo contará con capacidades básicas para el uso del sistema.

Factores Críticos de Éxito (FCE)

Tabla 19

Factor Crítico de Éxito

Factor de Éxito	Descripción	Criticidad
Definición clara del alcance del proyecto	El proyecto debe contar con un alcance claramente definido y aprobado para evitar cambios no controlados durante su ejecución.	Muy Alto
Cumplimiento del cronograma del proyecto	Las actividades deben ejecutarse dentro de los plazos establecidos para asegurar la entrega oportuna del proyecto.	Muy Alto
Control y seguimiento del proyecto	El proyecto debe contar con mecanismos de seguimiento que permitan verificar el avance y tomar acciones correctivas.	Muy Alto

Disponibilidad de recursos para el proyecto	El proyecto debe contar con los recursos humanos y tecnológicos necesarios durante todas sus fases.	Muy Alto

Riesgos

Tabla 20

Escala de probabilidad

Nivel	Categoría	Descripción cuantitativa	% de ocurrencia estimada
1	Imposible	Es altamente improbable que ocurra.	~0%
2	Poco probable	Podría ocurrir solo en circunstancias excepcionales.	10% – 25%
3	Probable	Puede ocurrir ocasionalmente.	25% – 50%
4	Muy probable	Es probable que ocurra varias veces.	50% – 75%
5	Seguro	Casi con certeza ocurrirá.	~100%

Tabla 21

Escala de impacto

Nivel	Categoría	Descripción del impacto
1	Insignificante	El riesgo no afecta o afecta mínimamente al proyecto.
2	Menor	Afecta parte del proyecto, pero puede manejarse fácilmente.
3	Moderado	Afecta el desarrollo normal y requiere medidas correctivas.
4	Significativo	Afecta objetivos clave del proyecto.
5	Crítico	Riesgo extremo que compromete el éxito del proyecto.

Tabla 22

Matriz de probabilidad e impacto

P \ I	1 - Insignificante	2 - Menor	3 - Moderado	4 - Significativo	5 - Crítico
1 - Imposible	1	2	3	4	5
2 - Poco probable	2	4	6	8	10
3 - Probable	3	6	9	12	15
4 - Muy probable	4	8	12	16	20
5 - Seguro	5	10	15	20	25

Tabla 23

Lista de Riegos

ID	Causa / Riesgo / Consecuencia	Calificación	Zona de Riesgo
R01	Debido a que el proyecto depende de un único responsable, existe el riesgo de que una enfermedad o indisposición detenga completamente el desarrollo, lo que generaría retrasos significativos o paralización temporal del proyecto.	**Muy Alta**	**Crítico**
R02	Debido a brechas de seguridad o configuraciones incorrectas, existe el riesgo de exposición o compromiso de datos de consumidores, lo que generaría sanciones legales, pérdida de confianza e impacto reputacional.	**Muy Alta**	**Crítico**
R03	Debido a la falta de presupuesto para mantenimiento, existe el riesgo de no poder actualizar ni corregir errores, lo que generaría un sistema obsoleto y con fallas críticas.	**Muy Alta**	**Crítico**
R04	Debido a la ausencia de procedimientos adecuados de respaldo, existe el riesgo de pérdida total o parcial de información, lo que impediría recuperar historiales y registros de pagos.	**Muy Alta**	**Crítico**
R05	Debido a cambios en la directiva o falta de continuidad organizacional, existe el riesgo de alteración en las prioridades del proyecto, lo que podría causar retrasos, abandono del sistema o ausencia de mantenimiento.	**Alta**	**Crítico**
R06	Debido a fallas en las pruebas técnicas, existe el riesgo de errores no detectados en cálculos o reportes, lo que podría provocar deudas mal calculadas, reclamos de usuarios y pérdida de confianza en el sistema.	**Alta**	**Crítico**
R07	Debido a requerimientos mal definidos, existe el riesgo de necesidad de rediseño durante el desarrollo, lo que incrementaría los tiempos y costos del proyecto.	**Alta**	**Crítico**
R08	Debido a una configuración inadecuada del sistema de roles, existe el riesgo de accesos indebidos a información sensible, lo que ocasionaría problemas legales y operativos.	**Alta**	**Alto**
R09	Debido a una integración deficiente entre backend y base de datos, existe el riesgo de fallos en la sincronización de deudas, lo que produciría discrepancias en montos y pérdida de confiabilidad.	**Alta**	**Alto**

R10	Debido al ingreso manual de datos, existe el riesgo de digitación errónea o duplicada, lo que generaría deudas incorrectas e inconsistencias en los recibos.	**Alta**	**Alto**
R11	Debido a la falta de capacitación del personal, existe el riesgo de un uso incorrecto del sistema, lo que ocasionaría errores en pagos, digitación equivocada y retrasos en los procesos administrativos.	**Media**	**Moderado**
R12	Debido a la dependencia del proveedor tecnológico, existe el riesgo de fallos o demoras en el soporte, lo que produciría interrupciones prolongadas del servicio.	**Media**	**Moderado**
R13	Debido a la baja calidad o inestabilidad del internet, existe el riesgo de interrupción en el acceso al sistema, lo que generaría registros incompletos de pagos y procesos administrativos pendientes.	**Media**	**Moderado**

Tabla 24

Estrategia para gestionar los riegos

ID	Riesgo	Estrategia de Gestión
R01	Dependencia de un único responsable → paralización total del proyecto	Designar un backup técnico o co-responsable; documentar procesos críticos y configuraciones; establecer un plan de continuidad operativa; capacitar a al menos un miembro adicional en despliegues y soporte; definir protocolos de emergencia para redistribuir tareas en caso de ausencia.
R02	Brechas de seguridad (configuración JWT, roles, etc.)	Aplicar buenas prácticas de seguridad en Spring Security; rotación periódica de claves JWT; validación estricta de roles y permisos; pruebas de penetración básicas; configuración de CORS y HTTPS; monitoreo de logs de acceso y alertas de intentos sospechosos.
R03	Dependencia de una sola persona para mantenimiento	Documentar configuraciones y despliegues; capacitar a más de un miembro técnico; definir un protocolo de soporte escalonado; crear manuales técnicos accesibles; establecer rotación de responsabilidades en el equipo.
R04	Inconsistencias en la base de datos	Definir modelo ER revisado; implementar constraints y triggers; usar migraciones con Flyway o Liquibase; validaciones backend antes de insertar datos; monitoreo de integridad con jobs automáticos; auditorías periódicas de registros críticos.
R05	Cambios en requisitos durante el desarrollo	Implementar gestión formal de cambios; documentar requisitos en un repositorio central; acordar un backlog fijo por sprint; validar cada incremento con el comité; usar herramientas de trazabilidad (Jira, Trello) para registrar cambios y su impacto.
R06	Errores no detectados en producción	Implementar pruebas unitarias, integrales y de aceptación; usar entornos separados (dev/test/prod); realizar QA por módulo crítico antes del despliegue; aplicar pruebas automatizadas en pipelines CI/CD; documentar casos de prueba críticos.
R07	Problemas por no seguir Clean Architecture	Revisiones semanales de arquitectura; checklist de buenas prácticas; separación estricta de capas; aplicar principios SOLID; pruebas unitarias por capa; code reviews obligatorios en cada merge request.
R08	Permisos mal asignados (riesgo en	Definir matriz de permisos; revisar roles antes del despliegue; crear pruebas específicas de autorización; monitorear intentos fallidos de acceso; aplicar

	roles)	autenticación multifactor (MFA); auditorías periódicas de accesos.
R09	Fallos en integración frontend-backend	Definir contratos API claros (OpenAPI/Swagger); pruebas e2e; integración continua; validaciones en DTOs; monitoreo de errores en tiempo real; pruebas de regresión en cada release.
R10	Problemas de conectividad en la localidad	Ofrecer sincronización offline donde sea posible; diseño ligero en frontend; usar servidores confiables con redundancia; permitir reintentos seguros de requests; monitorear disponibilidad de red; evaluar proveedores alternativos de internet.
R11	Uso inadecuado del sistema por falta de capacitación	Programar sesiones de capacitación para tesorero, secretario y administrador; crear manuales de usuario; realizar prácticas supervisadas antes del uso real; establecer un sistema de soporte interno para dudas frecuentes; evaluar competencias con simulaciones periódicas.
R12	Pérdida de información por falta de backups	Implementar backups automáticos diarios; pruebas periódicas de restauración; almacenar copias fuera del servidor principal; usar almacenamiento redundante en la nube; definir políticas de retención y recuperación.
R13	Retrasos en validación de entregables	Establecer calendario fijo de revisiones; avisos previos al comité; involucrar a más de una persona en la validación; usar herramientas colaborativas (Teams, Trello) para seguimiento; definir tiempos máximos de respuesta y escalamiento en caso de retraso.

Comunicaciones

Tabla 25

Tabla de comunicaciones

Comunicación	Descripción
Reuniones para el estado del proyecto	Reuniones periódicas (semanales o quincenales) donde el equipo de desarrollo se reúne con los interesados principales (presidente, tesorero, secretario) para revisar avances, bloquear problemas, revisar hitos y confirmar alineación con el alcance del sistema.
Informe de avances	Elaboración de un informe mensual con el progreso general, módulos completados, riesgos identificados y acciones realizadas. Este informe se envía a los interesados para mantener trazabilidad de la evolución del proyecto.
Comunicación directa con los interesados	Comunicación directa y continua, mediante reuniones o mensajería, para recibir retroalimentación sobre funcionalidades, aclarar requerimientos y gestionar sugerencias o cambios. Garantiza que el producto cumpla con las necesidades reales de la comunidad.
Revisión y validación de entregables	Sesiones formales de revisión cada vez que se completa un módulo. Los interesados prueban las funcionalidades y aprueban o solicitan ajustes antes de avanzar al siguiente ciclo.

Capítulo III:

Ejecución del Proyecto

Fase Inicio

Visión de Proyecto

La visión del proyecto es implementar un sistema digital integral que modernice y optimice la gestión administrativa y operativa del servicio de agua potable y alcantarillado del Centro Poblado Lurinchincha. El sistema permitirá automatizar los procesos de registro de consumidores, generación de deudas, control de pagos, gestión de servicios especiales y elaboración de reportes; garantizando información precisa, accesible y segura. El objetivo final es fortalecer la transparencia, reducir errores manuales, mejorar la eficiencia en la toma de decisiones y ofrecer a los usuarios un canal confiable donde puedan consultar sus deudas y movimientos. Con este proyecto, la JASS busca evolucionar hacia una gestión más ordenada, sostenible y orientada al servicio de la comunidad.

Presentación del Lanzamiento del Proyecto (Kick Off)

Historias de Usuarios (HU)

Las historias de usuario son descripciones breves y centradas en el usuario que representan funcionalidades específicas que el sistema debe ofrecer, redactadas en un lenguaje simple para asegurar la comprensión entre todos los interesados del proyecto. Su objetivo es definir qué necesita el usuario, con qué propósito y qué valor aporta al negocio, facilitando así la planificación y priorización dentro de metodologías ágiles como Scrum. Estas historias se estructuran generalmente bajo el formato: "Como [usuario], quiero [funcionalidad], para [beneficio]" (Scrum Guide, 2020).

Tabla 26

Historia de Usuarios

	HU	**Descripción de las Historias de Usuario (como, quiero, para)**
HU01	Registrar consumidores	Como secretario, quiero registrar nuevos consumidores, para mantener actualizada la base de usuarios que reciben el servicio.
HU02	Actualizar consumidores	Como secretario, quiero actualizar la información de los consumidores, para corregir datos o reflejar cambios como dirección o número de contacto.
HU03	Eliminar consumidores	Como administrador, quiero eliminar consumidores del sistema, para depurar registros o retirar usuarios que ya no reciben el servicio.
HU04	Registrar pagos mensuales	Como tesorero, quiero registrar los pagos mensuales de los consumidores, para mantener el control de los ingresos del servicio.

HU05	Verificar deudas anteriores	Como tesorero, quiero visualizar las deudas anteriores de un consumidor, para realizar el cobro correcto antes de registrar un nuevo pago.
HU06	Calcular deuda total	Como tesorero, quiero que la deuda total del consumidor se calcule automáticamente, para evitar errores manuales y agilizar el proceso de cobro.
HU07	Registrar pagos parciales	Como tesorero, quiero registrar pagos parciales de los consumidores, para permitir abonos cuando no puedan pagar el monto completo.

HU08	Consultar saldo pendiente	Como tesorero, quiero ver el saldo pendiente del consumidor después de un pago parcial, para continuar el seguimiento de la deuda.
HU09	Generar recibo	Como tesorero, quiero generar recibos de pago, para entregar comprobantes claros y verificables a los consumidores.
HU10	Ver historial de pagos	Como tesorero, quiero consultar el historial de pagos de un consumidor, para verificar transacciones anteriores y detectar inconsistencias.
HU11	Consumidor ve su historial	Como consumidor, quiero ver mi historial de pagos, para tener un control de lo que he pagado.
HU12	Reportes mensuales	Como tesorero, quiero generar reportes financieros mensuales, para evaluar los ingresos y movimientos del periodo.
HU13	Reportes anuales o por rango	Como administrador, quiero generar reportes anuales o por rango de fechas, para analizar la gestión y tomar decisiones administrativas.
HU14	Registrar roles	Como administrador, quiero gestionar los roles del sistema, para controlar qué acciones puede realizar cada usuario.
HU15	Asignar roles	Como administrador, quiero asignar roles a los usuarios, para garantizar que tengan los permisos adecuados a sus funciones (tesorero, secretario, etc.).
HU16	Iniciar sesión	Como usuario del sistema (consumidor externo o miembro del equipo administrativo), quiero iniciar sesión con mi usuario y contraseña, para acceder de forma segura a mi información personal y financiera, y a las funcionalidades que me corresponden según mi rol.
HU17	Consultar deuda desde el portal	Como consumidor, quiero consultar mi deuda total desde mi cuenta, para conocer cuánto debo pagar.
HU18	Registrar servicios especiales	Como tesorero, quiero registrar servicios especiales (corte, reconexión, mantenimiento), para que éstos impacten automáticamente en la deuda del consumidor.
HU19	Ver historial de servicios especiales	Como administrador, quiero visualizar los servicios especiales aplicados, para llevar un registro claro de acciones asociadas a cada usuario.

Product Backlog

El Product Backlog es una lista priorizada y evolutiva que contiene todas las funcionalidades, requerimientos, mejoras y correcciones que deben desarrollarse en un proyecto ágil. Representa la fuente única y centralizada de trabajo para el equipo Scrum y es gestionado

por el Product Owner, quien asegura que los elementos estén claramente definidos, ordenados por valor y alineados con los objetivos del producto. Su contenido incluye historias de usuario, tareas técnicas y cualquier necesidad del sistema, y se actualiza continuamente conforme avanza el proyecto o cambian las prioridades (Scrum Guide, 2020).

Tabla 27

Product Backlog

Épica	**HU ID**	**Historia de Usuario**
	HU01	Registrar consumidores
Gestión de Consumidores	HU02	Actualizar consumidores
	HU03	Eliminar consumidores
	HU04	Registrar pagos mensuales
	HU05	Verificar deudas anteriores
	HU06	Calcular automáticamente la deuda total
Gestión de Pagos y Deudas	HU07	Registrar pagos parciales
	HU08	Consultar saldo pendiente
	HU09	Generar recibos de pago
	HU10	Consultar historial de pagos (tesorero)
Portal del Consumidor	HU11	Consultar historial de pagos (consumidor)
	HU12	Generar reportes mensuales
Reportes Financieros	HU13	Generar reportes anuales o por rango de fechas
Gestión de Roles y Usuarios	HU14	Gestionar roles del sistema
	HU15	Asignar roles a los usuarios
	HU16	Iniciar sesión con usuario y contraseña
Autenticación y Seguridad	HU17	Consultar deuda desde el portal del consumidor
Servicios Especiales	HU18	Registrar servicios especiales (corte, reconexión, mantenimiento)
	HU19	Ver historial de servicios especiales

Fase Planificación

Historias de Usuarios Priorizados

Las historias de usuario priorizadas permiten organizar el Product Backlog según la importancia y el valor que cada funcionalidad aporta al proyecto. Esta priorización ayuda al Product Owner y al equipo a decidir qué se desarrollará primero y qué puede esperar, asegurando que el sistema entregue valor real en cada iteración. Para ello se emplean técnicas como MoSCoW, que clasifica las historias en “imprescindibles”, “deseables”, “opcionales” y “no consideradas

por ahora" (Cohn, 2004).

Figura 10

Método de MOSCOW

Nota: Tomado de El método MoSCoW (A5C53C2D01).

https://opencontent.upct.es/f0fc64e3b2d34f3a872a841a9c37b249/3f9bb933ba6440e59b0db2ecf6 5fc8a8/

Tabla 28

Historias de Usuarios Priorizadas

Épica	HU ID	Historia de Usuario	Prioridad MoSCoW
	HU01	Registrar consumidores	**M**
Gestión de Consumidores	HU02	Actualizar consumidores	**S**
	HU03	Eliminar consumidores	**S**
	HU04	Registrar pagos mensuales	**M**
	HU05	Verificar deudas anteriores	**M**
	HU06	Calcular automáticamente la deuda total	**M**
Gestión de Pagos y Deudas	HU07	Registrar pagos parciales	**S**
	HU08	Consultar saldo pendiente	**S**
	HU09	Generar recibos de pago	**M**
	HU10	Consultar historial de pagos (tesorero)	**S**
Portal del Consumidor	HU11	Consultar historial de pagos (consumidor)	**C**
Reportes Financieros	HU12	Generar reportes mensuales	**S**
	HU13	Generar reportes anuales o por rango de fechas	**C**
Gestión de Roles y Usuarios	HU14	Gestionar roles del sistema	**M**
	HU15	Asignar roles a los usuarios	**M**
Autenticación y Seguridad	HU16	Iniciar sesión con usuario y contraseña	**M**
	HU17	Consultar deuda desde el portal del consumidor	**C**

Servicios Especiales	HU18	Registrar servicios especiales (corte, reconexión, mantenimiento)	**S**
	HU19	Ver historial de servicios especiales	**C**

Estimación de Tareas de las HU

Para estimar el esfuerzo de cada Historia de Usuario se utiliza la técnica ágil **Planning Poker** , donde se asignan puntajes basados en la complejidad y el trabajo requerido usando una escala como Fibonacci. Esta dinámica permite que todos los involucrados aporten su percepción, se reduzcan sesgos y se logren estimaciones más realistas para planificar los sprints de manera efectiva. Es una de las técnicas más usadas por su simplicidad y precisión en equipos ágiles (Comparasoftware, 2021; Scrum.org, 2022).

Figura 11

Planning Poker

Nota: Tomado de Planning Poker: la técnica ágil para mejorar la estimación en Scrum.

https://www.appvizer.es/revista/organizacion-planificacion/gestion-proyectos/planning-poker

Figura 12

Aplicación Planning Poker

Figura 13

Captura de Planning Poker

Nota: Se ha utilizado Planning Poker mostrado en la figura 12 y 13 descargado de Play Store para la estimación de las Historias de Usuario.

Tabla 29

Estimación de tareas de HU

SPRINT	HU_ID	ID Tarea	VALOR	TOTAL
Sprint 1	HU01	T01 - Identificar los datos que deben registrarse para crear un consumidor.	2	25
		T02 - Definir los campos necesarios para el registro.	3	
		T03 - Diseñar el formulario para registrar consumidores.	5	
		T04 - Establecer las validaciones obligatorias del formulario.	3	
		T05 - Definir el flujo de registro de consumidores.	2	
		T06 - Construir la consulta necesaria para almacenar nuevos consumidores.	5	
		T07 - Validar que el registro se guarde correctamente.	3	
		T08 - Probar el flujo completo de registro.	2	
	HU02	T09 - Identificar qué datos pueden actualizarse del consumidor.	2	17
		T10 - Definir los campos editables y restricciones.	2	
		T11 - Diseñar el formulario de edición.	5	
		T12 - Definir el flujo de actualización de datos.	2	
		T13 - Construir la consulta para actualizar los datos.	3	
		T14 - Validar que los cambios se reflejen correctamente.	3	
	HU03	T15 - Definir el proceso para eliminar un consumidor.	2	13
		T16 - Establecer si la eliminación será lógica o permanente.	2	
		T17 - Diseñar la pantalla o confirmación para la eliminación.	3	
		T18 - Construir la operación para eliminar el registro.	3	
		T19 - Validar que el consumidor no aparezca después de ser eliminado.	3	
	HU14	T20 - Identificar los roles necesarios en el sistema.	2	11
		T21 - Definir los campos requeridos para registrar roles.	2	
		T22 - Diseñar el formulario de creación de roles.	3	
		T23 - Validar que los roles registrados puedan asignarse.	4	
	HU15	T24 - Identificar las acciones permitidas en cada rol.	3	16
		T25 - Definir el flujo para asignar roles.	3	
		T26 - Diseñar la pantalla para asignación de roles.	5	
		T27 - Validar que los permisos se asignen correctamente.	5	
	HU16	T28 - Identificar los datos necesarios para iniciar sesión.	3	24
		T29 - Diseñar el formulario de inicio de sesión.	3	
		T30 - Definir el flujo de autenticación.	5	
		T31 - Verificar credenciales y roles del usuario.	5	
		T32 - Validar mensajes de error ante accesos inválidos.	5	
		T33 - Probar el flujo completo de autenticación.	3	

Sprint 2	HU04	T34 - Identificar los datos necesarios para registrar un pago.	3	21
		T35 - Definir los campos del formulario de pago.	3	
		T36 - Diseñar la pantalla de registro de pagos.	5	
		T37 - Establecer validaciones para evitar pagos duplicados.	5	
		T38 - Construir la consulta para almacenar el pago.	3	
		T39 - Probar el registro de pagos completos.	2	
	HU05	T40 - Identificar qué información necesita el tesorero sobre deudas anteriores.	3	14
		T41 - Definir los datos a mostrar sobre deudas previas.	3	
		T42 - Diseñar la pantalla de consulta de deudas anteriores.	3	
		T43 - Construir la consulta que recupera deudas previas.	5	
	HU06	T44 - Definir cómo debe calcularse la deuda total.	3	16
		T45 - Identificar las fuentes necesarias para el cálculo.	3	
		T46 - Establecer reglas para sumar pagos y aplicar cargos.	5	
		T47 - Validar que el cálculo final sea correcto.	5	
Sprint 3	HU07	T48 - Identificar los datos necesarios para registrar un pago parcial.	3	16
		T49 - Diseñar el formulario de pagos parciales.	5	
		T50 - Definir cómo se calcula el saldo restante.	3	

		T51 - Validar que el saldo pendiente se actualice correctamente.	5	
	HU08	T52 - Identificar los datos necesarios para mostrar el saldo pendiente.	2	8
		T53 - Diseñar la interfaz que muestra el saldo pendiente.	3	
		T54 - Validar que el saldo esté actualizado.	3	
	HU09	T55 - Identificar los datos necesarios para generar un recibo.	2	18
		T56 - Diseñar el formato del recibo.	3	
		T57 - Definir el proceso de generación del recibo.	5	
		T58 - Validar que el recibo corresponda al pago realizado.	5	
		T59 - Probar la generación y descarga del recibo.	3	
		T60 - Identificar los datos necesarios para visualizar el historial de pagos.	2	
	HU10	T61 - Diseñar la vista del historial para el tesorero.	3	13
		T62 - Definir los datos que deben recuperarse del historial.	3	
		T63 - Validar orden cronológico y coherencia.	5	
	HU11	T64 - Identificar la información necesaria para el historial del consumidor.	2	10
		T65 - Diseñar la interfaz para ver el historial.	3	
		T66 - Validar que el usuario solo vea su propio historial.	3	
		T67 - Probar visualizaciones desde diferentes dispositivos.	2	

		T68 - Identificar qué datos necesita ver el consumidor sobre su deuda total.	2	
	HU17	T69 - Diseñar la pantalla que resume la deuda total.	3	11
		T70 - Validar que el cálculo de la deuda sea correcto.	3	
		T71 - Probar acceso desde el portal de consumidor.	3	
Sprint 4	HU12	T72 - Identificar qué datos se requieren para el reporte mensual.	2	16
		T73 - Diseñar filtros y estructura del reporte mensual.	3	
		T74 - Construir la consulta que obtiene datos mensuales.	5	
		T75 - Validar consistencia de información mensual.	3	
		T76 - Probar exportación del reporte mensual.	3	

		T77 - Identificar qué datos deben incluirse en reportes anuales o por rango.	2	18
	HU13	T78 - Definir filtros avanzados para el reporte.	3	
		T79 - Diseñar la interfaz para reportes anuales o por rango.	3	
		T80 - Construir consultas para rangos amplios de fechas.	5	
		T81 - Validar resultados y totales del reporte.	5	
	HU18	T82 - Identificar qué servicios especiales pueden registrarse.	2	18
		T83 - Definir los datos necesarios para registrar un servicio especial.	3	
		T84 - Diseñar el formulario para registrar servicios especiales.	3	
		T85 - Definir cómo afecta cada servicio a la deuda del consumidor.	5	
		T86 - Validar que el impacto en la deuda sea correcto.	5	
	HU19	T87 - Identificar la información que debe mostrar el historial de servicios especiales.	2	10
		T88 - Diseñar la vista del historial de servicios especiales.	3	
		T89 - Construir la consulta para recuperar historial.	3	
		T90 - Validar orden cronológico y consistencia.	2	

Sprint Planning

Tabla 30

Sprint Planning

Sprint 1	Sprint 2	Sprint 3	Sprint 4

Registrar consumidores	Registrar pagos mensuales	Registrar pagos parciales	Generar reportes mensuales
Actualizar consumidores	Verificar deudas anteriores	Consultar saldo pendiente	Generar reportes anuales o por rango de fechas
Eliminar consumidores	Calcular automáticamente la deuda total	Generar recibos de pago	Registrar servicios especiales (corte, reconexión, mantenimiento)
Gestionar roles del sistema		Consultar historial de pagos (tesorero)	Ver historial de servicios especiales
Asignar roles a los usuarios		Consultar historial de pagos (consumidor)	
Iniciar sesión con usuario y contraseña		Consultar deuda desde el portal del consumidor	

Descomposición de Tareas de las HU

Tabla 31

Descomposición de tareas

SPRINT	HU_ID	Tarea	Descripción	Estimación
Sprint 1	HU01	T01 - Identificar datos	Identificar los datos que deben registrarse para crear un consumidor.	2
		T02 - Definir campos	Definir los campos necesarios para el registro.	3
		T03 - Diseñar formulario	Diseñar el formulario para registrar consumidores.	5
		T04 - Validar formulario	Establecer las validaciones obligatorias del formulario.	3
		T05 - Definir flujo	Definir el flujo de registro de consumidores.	2
		T06 - Construir consulta	Construir la consulta necesaria para almacenar nuevos consumidores.	5
		T07 - Validar registro	Validar que el registro se guarde correctamente.	3
		T08 - Probar flujo completo	Probar el flujo completo de registro.	2
	HU02	T09 - Identificar datos editables	Identificar qué datos pueden actualizarse del consumidor.	2
		T10 - Definir campos editables	Definir los campos editables y restricciones.	2
		T11 - Diseñar formulario de edición	Diseñar el formulario de edición.	5

		T12 - Definir flujo de actualización	Definir el flujo de actualización de datos.	2
		T13 - Construir consulta de actualización	Construir la consulta para actualizar los datos.	3
		T14 - Validar actualización	Validar que los cambios se reflejen correctamente.	3
	HU03	T15 - Definir proceso de eliminación	Definir el proceso para eliminar un consumidor.	2
		T16 - Definir tipo de eliminación	Establecer si la eliminación será lógica o permanente.	2
		T17 - Diseñar pantalla de confirmación	Diseñar la pantalla o confirmación para la eliminación.	3
		T18 - Implementar eliminación	Construir la operación para eliminar el registro.	3
		T19 - Validar eliminación	Validar que el consumidor no aparezca después de ser eliminado.	3
	HU14	T20 - Identificar roles	Identificar los roles necesarios en el sistema.	2
		T21 - Definir campos de roles	Definir los campos requeridos para registrar roles.	2
		T22 - Diseñar formulario de roles	Diseñar el formulario de creación de roles.	3
		T23 - Validar roles	Validar que los roles registrados puedan asignarse.	4
	HU15	T24 - Identificar acciones por rol	Identificar las acciones permitidas en cada rol.	3
		T25 - Definir flujo de asignación	Definir el flujo para asignar roles.	3

		T26 - Diseñar pantalla de asignación	Diseñar la pantalla para asignación de roles.	5
		T27 - Validar permisos asignados	Validar que los permisos se asignen correctamente.	5
	HU16	T28 - Identificar datos de login	Identificar los datos necesarios para iniciar sesión.	3
		T29 - Diseñar formulario de login	Diseñar el formulario de inicio de sesión.	3
		T30 - Definir flujo de autenticación	Definir el flujo de autenticación.	5
		T31 - Verificar credenciales	Verificar credenciales y roles del usuario.	5
		T32 - Validar errores	Validar mensajes de error ante accesos inválidos.	5
		T33 - Probar autenticación	Probar el flujo completo de autenticación.	3
Sprint 2	HU04	T34 - Identificar datos para pagos	Identificar los datos necesarios para registrar un pago.	3
		T35 - Definir campos de pago	Definir los campos del formulario de pago.	3
		T36 - Diseñar formulario de pagos	Diseñar la pantalla de registro de pagos.	5
		T37 - Validar pagos duplicados	Establecer validaciones para evitar pagos duplicados.	5
		T38 - Construir consulta de pago	Construir la consulta para almacenar el pago.	3
		T39 - Probar registro de pagos	Probar el registro de pagos completos.	2
	HU05	T40 - Identificar datos de deudas anteriores	Identificar qué información necesita el tesorero sobre deudas anteriores.	3
		T41 - Definir datos a mostrar	Definir los datos a mostrar sobre deudas previas.	3

		T42 - Diseñar pantalla de deudas	Diseñar la pantalla de consulta de deudas anteriores.	3
		T43 - Construir consulta de deudas	Construir la consulta que recupera deudas previas.	5
	HU06	T44 - Definir cálculo de deuda total	Definir cómo debe calcularse la deuda total.	3
		T45 - Identificar fuentes del cálculo	Identificar las fuentes necesarias para el cálculo.	3
		T46 - Establecer reglas de cálculo	Establecer reglas para sumar pagos y aplicar cargos.	5
		T47 - Validar cálculo final	Validar que el cálculo final sea correcto.	5

Sprint 3	HU07	T48 - Identificar datos para pago parcial	Identificar los datos necesarios para registrar un pago parcial.	3
		T49 - Diseñar formulario de pagos parciales	Diseñar el formulario de pagos parciales.	5
		T50 - Definir cálculo de saldo restante	Definir cómo se calcula el saldo restante.	3
		T51 - Validar saldo pendiente	Validar que el saldo pendiente se actualice correctamente.	5
	HU08	T52 - Identificar datos del saldo	Identificar los datos necesarios para mostrar el saldo pendiente.	2
		T53 - Diseñar interfaz del saldo	Diseñar la interfaz que muestra el saldo pendiente.	3
		T54 - Validar saldo actualizado	Validar que el saldo esté actualizado.	3
	HU09	T55 - Identificar datos del recibo	Identificar los datos necesarios para generar un recibo.	2
		T56 - Diseñar formato de recibo	Diseñar el formato del recibo.	3
		T57 - Definir proceso del recibo	Definir el proceso de generación del recibo.	5
		T58 - Validar recibo generado	Validar que el recibo corresponda al pago realizado.	5
		T59 - Probar descarga del recibo	Probar la generación y descarga del recibo.	3
	HU10	T60 - Identificar datos del historial	Identificar los datos necesarios para visualizar el historial de pagos.	2
		T61 - Diseñar vista del historial	Diseñar la vista del historial para el tesorero.	3
		T62 - Definir datos del historial	Definir los datos que deben recuperarse del historial.	3
		T63 - Validar consistencia del historial	Validar orden cronológico y coherencia.	5
		T64 - Identificar información del historial del consumidor	Identificar la información necesaria para el historial del consumidor.	2
	HU11	T65 - Diseñar interfaz del historial	Diseñar la interfaz para ver el historial.	3
		T66 - Validar acceso restringido	Validar que el usuario solo vea su propio historial.	3
		T67 - Probar en varios dispositivos	Probar visualizaciones desde diferentes dispositivos.	2
	HU17	T68 - Identificar datos de deuda total	Identificar qué datos necesita ver el consumidor sobre su deuda total.	2

		T69 - Diseñar vista de deuda total	Diseñar la pantalla que resume la deuda total.	3

		T70 - Validar cálculo de deuda	Validar que el cálculo de la deuda sea correcto.	3
		T71 - Probar acceso desde portal	Probar acceso desde el portal de consumidor.	3
Sprint 4	HU12	T72 - Identificar datos para reporte mensual	Identificar qué datos se requieren para el reporte mensual.	2
		T73 - Diseñar filtros del reporte	Diseñar filtros y estructura del reporte mensual.	3
		T74 - Construir consulta mensual	Construir la consulta que obtiene datos mensuales.	5
		T75 - Validar datos mensuales	Validar consistencia de información mensual.	3
		T76 - Probar exportación mensual	Probar exportación del reporte mensual.	3
	HU13	T77 - Identificar datos para reporte anual	Identificar qué datos deben incluirse en reportes anuales o por rango.	2
		T78 - Definir filtros avanzados	Definir filtros avanzados para el reporte.	3
		T79 - Diseñar interfaz de reportes	Diseñar la interfaz para reportes anuales o por rango.	3
		T80 - Construir consulta por rango	Construir consultas para rangos amplios de fechas.	5
		T81 - Validar totales del reporte	Validar resultados y totales del reporte.	5
	HU18	T82 - Identificar servicios especiales	Identificar qué servicios especiales pueden registrarse.	2
		T83 - Definir datos del servicio especial	Definir los datos necesarios para registrar un servicio especial.	3
		T84 - Diseñar formulario de servicios especiales	Diseñar el formulario para registrar servicios especiales.	3
		T85 - Definir impacto en deuda	Definir cómo afecta cada servicio a la deuda del consumidor.	5
		T86 - Validar impacto en deuda	Validar que el impacto en la deuda sea correcto.	5
	HU19	T87 - Identificar datos del historial especial	Identificar la información que debe mostrar el historial de servicios especiales.	2
		T88 - Diseñar vista del historial especial	Diseñar la vista del historial de servicios especiales.	3
		T89 - Construir consulta del historial	Construir la consulta para recuperar historial.	3
		T90 - Validar consistencia del historial especial	Validar orden cronológico y consistencia.	2

Tabla 32

Plantilla Scrum

SPRINT	HU_ID	ID	Responsable	Inicio	Fin	Tiempo (horas)	Estado
Sprint 1	HU01	T01	Joseph Ipanaqué	01/12/2025	14/12/2025	1.51	Terminado
		T02	Joseph Ipanaqué	01/12/2025	14/12/2025	2.26	Terminado
		T03	Joseph Ipanaqué	01/12/2025	14/12/2025	3.77	Terminado
		T04	Joseph Ipanaqué	01/12/2025	14/12/2025	2.26	Terminado
		T05	Joseph Ipanaqué	01/12/2025	14/12/2025	1.51	Terminado
		T06	Joseph Ipanaqué	01/12/2025	14/12/2025	3.77	Terminado
		T07	Joseph Ipanaqué	01/12/2025	14/12/2025	2.26	Terminado
		T08	Joseph Ipanaqué	01/12/2025	14/12/2025	1.51	Terminado
	HU02	T09	Joseph Ipanaqué	01/12/2025	14/12/2025	1.51	Terminado
		T10	Joseph Ipanaqué	01/12/2025	14/12/2025	1.51	Terminado
		T11	Joseph Ipanaqué	01/12/2025	14/12/2025	3.77	Terminado
		T12	Joseph Ipanaqué	01/12/2025	14/12/2025	1.51	Terminado
		T13	Joseph Ipanaqué	01/12/2025	14/12/2025	2.26	Terminado
		T14	Joseph Ipanaqué	01/12/2025	14/12/2025	2.26	Terminado
	HU03	T15	Joseph Ipanaqué	01/12/2025	14/12/2025	1.51	Terminado
		T16	Joseph Ipanaqué	01/12/2025	14/12/2025	1.51	Terminado
		T17	Joseph Ipanaqué	01/12/2025	14/12/2025	2.26	Terminado
		T18	Joseph Ipanaqué	01/12/2025	14/12/2025	2.26	Terminado
		T19	Joseph Ipanaqué	01/12/2025	14/12/2025	2.26	Terminado
	HU14	T20	Joseph Ipanaqué	01/12/2025	14/12/2025	1.51	Terminado
		T21	Joseph Ipanaqué	01/12/2025	14/12/2025	1.51	Terminado
		T22	Joseph Ipanaqué	01/12/2025	14/12/2025	2.26	Terminado
		T23	Joseph Ipanaqué	01/12/2025	14/12/2025	3.02	Terminado
	HU15	T24	Joseph Ipanaqué	01/12/2025	14/12/2025	2.26	Terminado
		T25	Joseph Ipanaqué	01/12/2025	14/12/2025	2.26	Terminado
		T26	Joseph Ipanaqué	01/12/2025	14/12/2025	3.77	Terminado
		T27	Joseph Ipanaqué	01/12/2025	14/12/2025	3.77	Terminado
		T28	Joseph Ipanaqué	01/12/2025	14/12/2025	2.26	Terminado
	HU16	T29	Joseph Ipanaqué	01/12/2025	14/12/2025	2.26	Terminado
		T30	Joseph Ipanaqué	01/12/2025	14/12/2025	3.77	Terminado
		T31	Joseph Ipanaqué	01/12/2025	14/12/2025	3.77	Terminado
		T32	Joseph Ipanaqué	01/12/2025	14/12/2025	3.77	Terminado
		T33	Joseph Ipanaqué	01/12/2025	14/12/2025	2.26	Terminado

Sprint 2	HU04	T34	Joseph Ipanaqué	15/12/2025	28/12/2025	4.71	Terminado
		T35	Joseph Ipanaqué	15/12/2025	28/12/2025	4.71	Terminado
		T36	Joseph Ipanaqué	15/12/2025	28/12/2025	7.84	Terminado
		T37	Joseph Ipanaqué	15/12/2025	28/12/2025	7.84	Terminado
		T38	Joseph Ipanaqué	15/12/2025	28/12/2025	4.71	Terminado
		T39	Joseph Ipanaqué	15/12/2025	28/12/2025	3.14	Terminado
	HU05	T40	Joseph Ipanaqué	15/12/2025	28/12/2025	4.71	Terminado
		T41	Joseph Ipanaqué	15/12/2025	28/12/2025	4.71	Terminado
		T42	Joseph Ipanaqué	15/12/2025	28/12/2025	4.71	Terminado
		T43	Joseph Ipanaqué	15/12/2025	28/12/2025	7.84	Terminado
	HU06	T44	Joseph Ipanaqué	15/12/2025	28/12/2025	4.71	Terminado
		T45	Joseph Ipanaqué	15/12/2025	28/12/2025	4.71	Terminado
		T46	Joseph Ipanaqué	15/12/2025	28/12/2025	7.84	Terminado
		T47	Joseph Ipanaqué	15/12/2025	28/12/2025	7.84	Terminado
		T48	Joseph Ipanaqué	29/12/2025	11/01/2026	3.16	Terminado
Sprint 3	HU07	T49	Joseph Ipanaqué	29/12/2025	11/01/2026	5.26	Terminado
		T50	Joseph Ipanaqué	29/12/2025	11/01/2026	3.16	Terminado
		T51	Joseph Ipanaqué	29/12/2025	11/01/2026	5.26	Terminado
	HU08	T52	Joseph Ipanaqué	29/12/2025	11/01/2026	2.11	Terminado
		T53	Joseph Ipanaqué	29/12/2025	11/01/2026	3.16	Terminado
		T54	Joseph Ipanaqué	29/12/2025	11/01/2026	3.16	Terminado
	HU09	T55	Joseph Ipanaqué	29/12/2025	11/01/2026	2.11	Terminado
		T56	Joseph Ipanaqué	29/12/2025	11/01/2026	3.16	Terminado
		T57	Joseph Ipanaqué	29/12/2025	11/01/2026	5.26	Terminado
		T58	Joseph Ipanaqué	29/12/2025	11/01/2026	5.26	Terminado
		T59	Joseph Ipanaqué	29/12/2025	11/01/2026	3.16	Terminado
	HU10	T60	Joseph Ipanaqué	29/12/2025	11/01/2026	2.11	Terminado
		T61	Joseph Ipanaqué	29/12/2025	11/01/2026	3.16	Terminado
		T62	Joseph Ipanaqué	29/12/2025	11/01/2026	3.16	Terminado
		T63	Joseph Ipanaqué	29/12/2025	11/01/2026	5.26	Terminado
		T64	Joseph Ipanaqué	29/12/2025	11/01/2026	2.11	Terminado
	HU11	T65	Joseph Ipanaqué	29/12/2025	11/01/2026	3.16	Terminado
		T66	Joseph Ipanaqué	29/12/2025	11/01/2026	3.16	Terminado
		T67	Joseph Ipanaqué	29/12/2025	11/01/2026	2.11	Terminado
	HU17	T68	Joseph Ipanaqué	29/12/2025	11/01/2026	2.11	Terminado

		T69	Joseph Ipanaqué	29/12/2025	11/01/2026	3.16	Terminado
		T70	Joseph Ipanaqué	29/12/2025	11/01/2026	3.16	Terminado
		T71	Joseph Ipanaqué	29/12/2025	11/01/2026	3.16	Terminado
Sprint 4	HU12	T72	Joseph Ipanaqué	12/01/2026	25/01/2026	2.58	Terminado

		T73	Joseph Ipanaqué	12/01/2026	25/01/2026	3.87	Terminado
		T74	Joseph Ipanaqué	12/01/2026	25/01/2026	6.45	Terminado
		T75	Joseph Ipanaqué	12/01/2026	25/01/2026	3.87	Terminado
		T76	Joseph Ipanaqué	12/01/2026	25/01/2026	3.87	Terminado
	HU13	T77	Joseph Ipanaqué	12/01/2026	25/01/2026	2.58	Terminado
		T78	Joseph Ipanaqué	12/01/2026	25/01/2026	3.87	Terminado
		T79	Joseph Ipanaqué	12/01/2026	25/01/2026	3.87	Terminado
		T80	Joseph Ipanaqué	12/01/2026	25/01/2026	6.45	Terminado
		T81	Joseph Ipanaqué	12/01/2026	25/01/2026	6.45	Terminado
	HU18	T82	Joseph Ipanaqué	12/01/2026	25/01/2026	2.58	Terminado
		T83	Joseph Ipanaqué	12/01/2026	25/01/2026	3.87	Terminado
		T84	Joseph Ipanaqué	12/01/2026	25/01/2026	3.87	Terminado
		T85	Joseph Ipanaqué	12/01/2026	25/01/2026	6.45	Terminado
		T86	Joseph Ipanaqué	12/01/2026	25/01/2026	6.45	Terminado
	HU19	T87	Joseph Ipanaqué	12/01/2026	25/01/2026	2.58	Terminado
		T88	Joseph Ipanaqué	12/01/2026	25/01/2026	3.87	Terminado
		T89	Joseph Ipanaqué	12/01/2026	25/01/2026	3.87	Terminado
		T90	Joseph Ipanaqué	12/01/2026	25/01/2026	1.29	Terminado

Fase Ejecución

Incremento del Sprint Backlog

Durante el Sprint, las tareas de la historia de las historias de usuario se realizarán según el cronograma acordado. Este progreso se mostrará en el tablero de Trello.

Figura 14

Incremento del Sprint Backlog – Sprint 1

Figura 15

Incremento del Sprint Backlog – Sprint 2

Figura 16

Incremento del Sprint Backlog – Sprint 3

Figura 17

Incremento del Sprint Backlog – Sprint 4

Mantenimiento del Product Backlog

Figura 18

Mantenimiento del Product Backlog

Dayli Scrum Meeting

Figura 19

Dayli Scrum Meeting

Desarrollo del Spring Backlog

Figura 20

Desarrollo del Spring Backlog

Fase de Revisión y Retrospectiva

Scrum Board

Es una herramienta visual clave que el Equipo Scrum utiliza para organizar, planificar y monitorear el progreso de las tareas durante cada sprint. Dividido en columnas que representan diferentes etapas del trabajo, como "Pendiente", "Proceso," y "Terminado," permite que el equipo vea claramente el estado de cada tarea y las mueva a medida que avanzan

(SCRUMstudy™,2022).

Tabla 33

Scrum Board: Sprint 1

	SPRINT 1	
HU Por Hacer	**En Proceso**	**Terminados**
HU 01		T 01 - T 08
HU 02		T 09 - T 14
HU 03		T 15 - T 19
HU 14		T 20 - T 23
HU 15		T 24 - T 27
HU 16		T 28 - T 33

Tabla 34

Scrum Board: Sprint 2

	SPRINT 2	
HU Por Hacer	**En Proceso**	**Terminados**
HU 04		T 34 - T 39
HU 05		T 40 - T 43
HU 06		T 44 - T 47

Tabla 35
Scrum Board: Sprint 3

		SPRINT 3	
HU	**Por Hacer**	**En Proceso**	**Terminados**
HU 07			T 48 - T 51
HU 08			T 52 - T 54
HU 09			T 55 - T 59
HU 10			T 60 - T 63
HU 11			T 64 - T 67
HU 17			T 68 - T 71

Tabla 36
Scrum Board: Sprint 4

		SPRINT 4	
HU	**Por Hacer**	**En Proceso**	**Terminados**
HU 12			T 72 - T 76
HU 13			T 77 - T 81
HU 18			T 82 - T 86
HU 19			T 87 - T 90

Validación de los Sprint (Sprint Review)

Tabla 37

Sprint Review de Sprint 1, 2, 3 y 4

SPRINT	HU_ID	DESCRIPCIÓN	ESTADO
	HU01	Registrar consumidores	TERMINADO
	HU02	Actualizar consumidores	TERMINADO
Sprint 1	HU03	Eliminar consumidores	TERMINADO
	HU14	Gestionar roles del sistema	TERMINADO
	HU15	Asignar roles a los usuarios	TERMINADO
	HU16	Iniciar sesión con usuario y contraseña	TERMINADO
	HU04	Registrar pagos mensuales	TERMINADO
Sprint 2	HU05	Verificar deudas anteriores	TERMINADO
	HU06	Calcular automáticamente la deuda total	TERMINADO

Sprint 3 HU07
Registrar pagos parciales
TERMINADO
HU08
Consultar saldo pendiente
TERMINADO
HU10
Consultar historial de pagos (tesorero)
TERMINADO
HU17
Consultar deuda desde el portal del consumidor
TERMINADO
HU09
Generar recibos de pago
TERMINADO

HU11	Consultar historial de pagos (consumidor)	TERMINADO	

Sprint 4	HU12	Generar reportes mensuales	TERMINADO	
	HU13	Generar reportes anuales o por rango de fechas	TERMINADO	
	HU18	Registrar servicios especiales (corte, reconexión, mantenimiento)	TERMINADO	
	HU19	Ver historial de servicios especiales	TERMINADO	

Burn Down Chart

Figura 21

Burn Down Chart – Sprint 1

Figura 22

Burn Down Chart – Sprint 2

Figura 23

Burn Down Chart – Sprint 3

Figura 24

Burn Down Chart – Sprint 4

Creación de la lista de impedimento de log

Tabla 38

Impedimento de log

Código	Descripción	Estado
I01	Curva de aprendizaje en la implementación de la arquitectura Clean Architecture aplicada al backend.	Resuelto
I02	Conocimiento intermedio en la configuración de seguridad y autenticación usando Spring Security con JWT.	Resuelto
I03	Complejidad en el diseño de una base de datos robusta para la gestión de pagos, recibos y control de deudas.	Resuelto
I04	Falta de experiencia inicial en el desarrollo de interfaces web responsivas utilizando Angular.	Resuelto

Retrospectivas (sprint y del Product Backlog) Tabla 39

Retrospective del Sprint

Sprint	¿Qué se hizo bien en la iteración?	¿Qué no se hizo bien en la iteración?	¿Qué mejoras se implementará en la siguiente iteración?
SPRINT 1	Se definió correctamente la arquitectura del proyecto basada en Clean Architecture y se implementó la estructura inicial de módulos (roles, usuarios, consumidores). El equipo logró comprender la separación de capas y responsabilidades.	Se presentaron dificultades iniciales en la configuración de Spring Security y JWT debido a la complejidad del control de roles y permisos.	Fortalecer el conocimiento en Spring Security y documentar la configuración de seguridad para facilitar futuras implementaciones.

SPRINT 2	Se avanzó en el desarrollo de los casos de uso del sistema, incluyendo la gestión de pagos, recibos y servicios especiales, cumpliendo con los objetivos planificados.	La coordinación de horarios del equipo dificultó la resolución inmediata de dudas técnicas durante el desarrollo.	Mejorar la planificación de reuniones y definir horarios fijos para la revisión de avances y resolución de problemas.
SPRINT 3	Se logró refinar el modelo de base de datos en PostgreSQL, optimizando las relaciones y reglas de negocio para el cálculo de deuda y numeración de recibos. El trabajo se completó dentro del tiempo establecido.	Se invirtió tiempo adicional en refactorizar código debido a ajustes en buenas prácticas y mejoras en la organización de paquetes.	Aplicar estándares de codificación desde el inicio y reforzar el uso de buenas prácticas para reducir retrabajos.
SPRINT 4	Se consolidó la integración entre la lógica de negocio y la persistencia de datos, dejando el backend preparado para su consumo desde el frontend en Angular.	Aún no se cuenta con una interfaz gráfica funcional para validar completamente la experiencia del usuario final.	Iniciar el desarrollo del frontend en Angular, priorizando un diseño responsivo y una comunicación eficiente con el backend.

Pruebas de Aceptación

Tabla 40

Pruebas de Aceptación N° 1

Prueba de Aceptación N° 1 HISTORIA DE USUARIO HU16: Iniciar sesión

Como usuario del sistema (consumidor externo o miembro del equipo administrativo),

Descripción de la Historia

quiero iniciar sesión con mi usuario y contraseña, para acceder de forma segura a mi información personal y financiera, y a las funcionalidades que me corresponden según mi rol.

DEMO DE LA HISTORIA DE USUARIO Pantalla de Inicio de Sesión

Pantalla para usuario de la organización (interno)

Pantalla para usuario de la organización (externo)

Tabla 41

Pruebas de Aceptación N° 2

Prueba de Aceptación N° 2 HISTORIA DE USUARIO HU01: Registrar consumidores Descripción de la Historia
Como secretario, quiero registrar nuevos consumidores, para mantener actualizada la base de usuarios que reciben el servicio.

DEMO DE HISTORIA DE USUARIO

Tabla 42

Pruebas de Aceptación N° 3

Prueba de Aceptación N° 3 HISTORIA DE USUARIO HU14: Registrar roles Descripción de la Historia
Como administrador, quiero gestionar los roles del sistema, para controlar qué acciones puede realizar cada usuario.

DEMO DE HISTORIA DE USUARIO

88

Tabla 43

Pruebas de Aceptación N° 4

Prueba de Aceptación N° 4 HISTORIA DE USUARIO HU18: Registrar servicios especiales

100

Descripción de la Historia

Como tesorero, quiero registrar servicios especiales (corte, reconexión, mantenimiento), para que éstos impacten automáticamente en la deuda del consumidor.

DEMO DE HISTORIA DE USUARIO

Fase Transición y Cierre

Actas de Conformidad de los Entregables

Tabla 44

Acta de conformidad de los entregables

Acta de Conformidad del Proyecto

Nombre del Proyecto Siglas del Proyecto

DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA WEB PARA OPTIMIZAR LA GESTIÓN DE COBROS Y REPORTES POR SERVICIOS

DE AGUA POTABLE Y ALCANTARILLADO PARA LA JUNTA ADMINISTRADORA DE SERVICIO Y SANEAMIENTO (JASS) DEL CENTRO POBLADO LURINCHINCHA EN EL DEPARTAMENTO DE ICA DURANTE EL AÑO 2026

SISWEB-JASS

Nombre del Cliente o Sponsor Junta Administradora de Servicio y Saneamiento

Declaración de Aceptación Formal

Por medio de la presente, se declara que el Proyecto SISWEB-JASS ha sido concluido y entregado con éxito a la JASS – Junta Administradora de Servicio y Saneamiento, luego de haber sido sometido a una verificación exhaustiva. Durante la revisión se confirmó que todos los objetivos y requerimientos establecidos en el proyecto fueron alcanzados de manera satisfactoria y conforme a los criterios definidos. La JASS – Junta Administradora de Servicio y Saneamiento, en su calidad de cliente y entidad receptora, acepta y aprueba formalmente el proyecto, certificando así la finalización exitosa y la conformidad de la entrega.

Observaciones Adicionales Ninguna

Aceptado Por

Nombre del Cliente Fecha

Junta Administradora de Servicio y Saneamiento 06/02/2026 **Distribuido y Aceptado**

Presidente de la JASS

Capítulo IV:

Programación

Implementación de la Arquitectura de Software

Como se mencionó en la sección correspondiente a la arquitectura de software, el proyecto se implementa bajo un modelo cliente-servidor, y el backend adopta el enfoque de Clean Architecture mediante Scaffold Clean Architecture para la organización interna del software, lo cual se representa en el siguiente diagrama.:

Figura 21

Clean Architecture

Nota: Tomado de Scaffold Clean Architecture (https://bancolombia.github.io/scaffold-clean-architecture/docs/intro)

Creación de la Base de Datos

La base de datos del proyecto se denomina JASS_DB y fue implementada en PostgreSQL, estando compuesta por 12 entidades principales que soportan la gestión de consumidores, facturación, pagos, recibos, servicios especiales y control de deuda.

Figura 22

Modelo de base de datos

Implementación de Librerías y Dependencias

Tabla 45

Librerías y Dependencias Utilizadas

N°	Dependencia / Librería	Versión	Utilidad
1	Java	17	Lenguaje principal utilizado para el desarrollo del backend del sistema.
2	Spring Boot	3.x	Framework que facilita el desarrollo de aplicaciones empresariales basadas en Java, permitiendo la creación de APIs REST de forma rápida y estructurada.
3	Spring WebFlux	3.x	Proporciona soporte para programación reactiva y la exposición de servicios REST no bloqueantes.
4	Spring Security	6.x	Gestiona la autenticación y autorización del sistema mediante mecanismos de seguridad basados en JWT.
5	Spring Data R2DBC	3.x	Permite el acceso reactivo a la base de datos PostgreSQL desde el backend.
6	PostgreSQL	15	Sistema gestor de base de datos relacional utilizado para el almacenamiento de la información del sistema.
7	JSON Web Token (JWT)	—	Mecanismo utilizado para asegurar la comunicación entre el frontend y el backend mediante tokens.
8	Lombok	1.18.x	Reduce el código repetitivo en las clases Java, facilitando la generación automática de métodos comunes.

9	MapStruct	1.5.x	Herramienta utilizada para el mapeo entre entidades, modelos de dominio y DTOs.
10	Maven	3.x	Herramienta de gestión de dependencias y construcción del proyecto backend.
11	Angular	17	Framework utilizado para el desarrollo del frontend del sistema, basado en componentes y arquitectura modular.
12	TypeScript	5.x	Lenguaje utilizado en el frontend que permite un desarrollo más robusto y mantenible.
13	@angular/router	17	Permite la navegación entre vistas del sistema mediante rutas definidas.
14	@angular/forms	17	Facilita la creación y validación de formularios utilizados en la interfaz del usuario.
15	@angular/common/http	17	Proporciona el cliente HTTP para la comunicación entre el frontend y el backend mediante servicios REST.
16	RxJS	7.x	Biblioteca para la gestión de flujos de datos asíncronos en la aplicación frontend.
17	HTML5 / CSS3	—	Tecnologías utilizadas para la estructura y el diseño visual de la interfaz del sistema.
18	Flexbox / Grid	—	Utilizadas para garantizar un diseño responsivo adaptable a dispositivos móviles y de escritorio.

Codificación del Back End

El término Back-End "es utilizado para referirse al área lógica de toda página web. Nos referimos a la arquitectura interna del sitio que asegura que todos elementos desarrollen la función correcta. No está visible a ojos del usuario y no incluye ningún tipo de elemento gráfico" (Nestrategia, s.f.).

Figura 23

Codificación de back-end de inicio de sesión

Figura 24

Codificación de back-end de creación de consumidor

Figura 25

Codificación de back-end de creación de roles

Figura 26

Codificación de back-end de creación de servicio especiales

Se dará mas detalles sobre la codificación back-end en el anexo 2.

Codificación del Front End

El término Front-End "es la parte que ve el usuario y en la que sí se incluyen, al contrario que en Back-End, la línea de diseño y los elementos gráficos de la página. De ahí que su nombre sea Front (Parte frontal: la parte que sí se ve). Será aquí donde se incluyan los estilos, los colores, los fondos, tamaños y las animaciones del sitio web." (Nestrategia, s.f.).

Figura 27

Codificación de front-end de inicio de sesión

Figura 28

Codificación de front-end de creación del consumidor

Figura 29

Codificación de front-end de creación de roles

Figura 30

Codificación de front-end de creación de servicios especiales

Codificación de Consultas y Reportes Figura 31

Codificación de filtro de búsqueda de consumidores

Figura 32

Codificación de filtro de búsqueda de usuarios por dni

Codificación de Mantenedores (CRUD) Figura 33

Codificación de creación de servicio especial

Figura 34

Codificación de actualizar servicio especial

Figura 35

Codificación de listar todos los servicios especiales

Figura 36

Codificación de eliminar servicio especial

Codificación de Procesos Transaccionales

Figura 36

Codificación de método que sincroniza pagos con servicios asignado al consumidor

Capítulo V:

Pruebas de Calidad de Software

Pruebas Unitarias

Para las pruebas unitarias se hicieron TEST a los servicios REST(API) del Backend usando la herramienta POSTMAN, el servidor devuelve estados HTTP para comprobar errores y brindando mensajes en el cuerpo de la respuesta. De este modo se puede comprobar la funcionalidad del sistema.

Tabla 46

Respuestas HTTP

ESTADO	CÓDIGO
OK	200
BAD REQUEST	400
UNAUTHORIZED	401
FORBIDDEN	403
NOT FOUND	404
CONFLICT	409
INTERNAL SERVER ERROR	500

Tabla 47

Métodos HTTP

TIPO	DESCRIPCIÓN
GET	Se usa para obtener o leer datos del servidor. No modifica nada, solo recupera información.
POST	Se usa para **crear** un nuevo recurso en el servidor.
PUT	Se usa para **actualizar completamente** un recurso existente (reemplaza todo el objeto).
PATCH	Se usa para **actualizar parcialmente** un recurso (solo modifica los campos enviados).
DELETE	Se usa para **eliminar** un recurso del servidor.

Tabla 48

Prueba de inicio de sesión

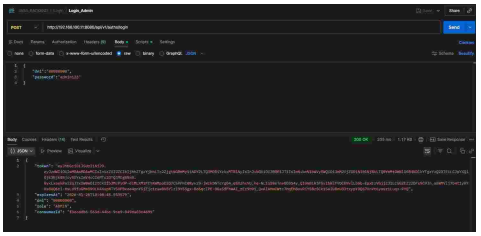
PRUEBA		Inicio de Sesión	
DESCRIPCIÓN DE LA PRUEBA	Consumir servicio Inicio de Sesión		
RESULTADO ESPERADO		Ingresar credenciales del USUARIO guardado en la base de datos, al enviar la petición se recibirá un token encriptado y el estado HTTP 200	
TIPO METODO	POST	**RESULTADO**	PASO
EVIDENCIAS			
			

Tabla 49

Prueba de Listado de Consumidores

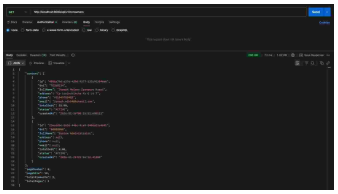
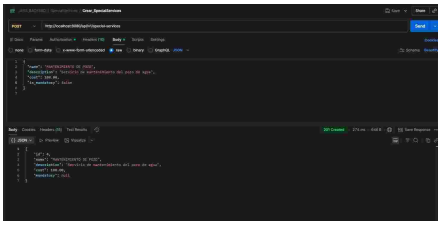
PRUEBA		Listado de Consumidores	
DESCRIPCIÓN DE LA PRUEBA	Consumir el servicio de listado de consumidores para obtener todos los registros disponibles (con paginación si aplica).		
RESULTADO ESPERADO		Al enviar la petición con un token válido, el servicio debe devolver: · Un listado de consumidores (en formato JSON) · Con el estado HTTP 200 OK · Y el cuerpo debe contener los datos esperados (id, nombre, DNI, estado, etc.)	
TIPO METODO	GET	**RESULTADO**	PASO
EVIDENCIAS			
			

Tabla 50

Prueba de Creación de servicio especial

PRUEBA		Creación de servicio especial	
DESCRIPCIÓN DE LA PRUEBA	Crear un servicio especial para la organización con datos válidos.		
RESULTADO ESPERADO		Se debe guardar el servicio y retornar 201 CREATED con el objeto creado.	
TIPO METODO	POST	**RESULTADO**	PASO

EVIDENCIAS	
	

Pruebas Integrales

Figura 37

Crear Consumidor

Figura 38

Modal de confirmación de creación de consumidor

Nota. El código generado en la creación de usuario es usado para crear su usuario y contraseña.

Figura 39

Listado de consumidores

Nota. Se observa que al momento de crear se le genero una deuda al consumidor por los servicios obligatorios que se deben pagar.

Figura 40

Pantalla de detalles de servicios asignado al consumidor

Nota. Servicios asignados al consumidor de manera automática al momento de su creación.

Figura 41

Pantalla de detalle del pago que deberá pagar el consumidor del periodo actual

Nota. Se evidencia el cobro de los servicios especiales que se le asigno al consumidor al momento de su creación.

Conclusiones

Conclusión 1

El análisis del negocio permitió identificar claramente las limitaciones actuales de la JASS del Centro Poblado Lurinchincha, evidenciando procesos manuales, falta de trazabilidad y ausencia de herramientas tecnológicas para el control de deudas, pagos y reportes financieros. La evaluación de la organización, del entorno operativo y del FODA confirmó la necesidad de implementar un sistema web que unifique la información, reduzca errores y fortalezca la transparencia en la gestión del servicio de agua potable y alcantarillado.

Conclusión 2

La planificación del proyecto estableció una estructura sólida para el desarrollo del sistema, definiendo un enfoque ágil basado en Scrum, una arquitectura escalable y módulos precisos alineados a los requerimientos levantados en el análisis. Asimismo, la declaración de alcance, los objetivos, los factores críticos de éxito y la identificación de interesados garantizan que el proyecto tenga una dirección clara y una coordinación efectiva para su ejecución, reduciendo riesgos y asegurando que la solución propuesta responda adecuadamente a las necesidades de la JASS.

Conclusión 3

El análisis del proceso de desarrollo y gestión del proyecto evidenció la necesidad de contar con herramientas que permitan mejorar la eficiencia, calidad y control del trabajo técnico y administrativo. En este contexto, la incorporación de inteligencia artificial se presenta como un apoyo estratégico para automatizar tareas repetitivas, fortalecer la calidad del software y

optimizar la planificación y seguimiento del proyecto, contribuyendo a una ejecución más ordenada, precisa y orientada a resultados.

Conclusión 4

La evaluación de los riesgos asociados al manejo de información sensible y a la operación del sistema permitió identificar la importancia de establecer mecanismos de seguridad desde las etapas iniciales del desarrollo. La correcta definición de roles, permisos y controles de acceso resulta fundamental para proteger los datos de los consumidores y del equipo administrativo, así como para garantizar la integridad, confidencialidad y trazabilidad de las operaciones realizadas dentro del sistema.

Recomendaciones

Recomendación 1

Se recomienda mantener un proceso continuo de revisión y actualización de los requisitos funcionales y no funcionales, involucrando activamente a los usuarios clave (presidente, tesorero, secretario). Esto permitirá asegurar que el sistema responda a las necesidades reales de la organización y evitar retrabajos durante la fase de desarrollo.

Recomendación 2

Se recomienda establecer un control estricto del cronograma y reuniones periódicas con los interesados para revisar avances, riesgos y cambios potenciales. Asimismo, documentar cada entrega o incremento asegurará que la planificación siga alineada con los objetivos estratégicos y que el proyecto se mantenga dentro del alcance, tiempo y recursos definidos.

Recomendación 3

Se recomienda usar la inteligencia artificial para optimizar el desarrollo y la gestión de proyectos. El uso de IA ayuda en tareas como generación de código, revisión automática de calidad, detección temprana de errores, estimación de tiempos y gestión de tareas. Esto permite acelerar el desarrollo, mejorar la calidad del software y facilitar la toma de decisiones basada en datos, aumentando la productividad del equipo y reduciendo el riesgo de retrasos.

Recomendación 4

Se recomienda implementar un sistema robusto de control de acceso y seguridad desde las primeras etapas del desarrollo. Esto incluye la definición clara de roles y permisos, autenticación segura (por ejemplo, JWT), validación de entradas y registro de auditoría de acciones críticas. Garantizar la seguridad desde el inicio evita vulnerabilidades en producción y protege la información sensible de los consumidores y del equipo administrativo.

Referencias Bibliográficas

González, M. (2023). Recursos tecnológicos en el entorno educativo y empresarial. Revista de Innovación y Tecnología, 12(3), 45–58.

Ishikawa, K. (1990). Introduction to Quality Control. Chapman & Hall.

Raeburn, A. (2025). Factibilidad técnica: qué es, criterios y análisis. Lifeder. https://www.lifeder.com/factibilidad-tecnica/

Shopify. (2025). What is a feasibility study? Benefits, types, and examples. https://www.shopify.com/blog/feasibility-study

SCRUMstudy™. (2022). *A Guide to the Scrum Body of Knowledge (SBOK™ Guide)* (4th ed.).

SCRUMstudy™. (2017). Guía de los Fundamentos de SCRUM (Guía SBOK®) (3ra ed.).

SCRUMstudy

Masood, Z., Hoda, R., & Blincoe, K. (2021). Real world Scrum: A grounded theory of variations in practice. *arXiv* . https://arxiv.org/abs/2103.15268

Sommerville, I. (2016). *Software Engineering* (10th ed.). Pearson. Google Developers. (2023). *Angular documentation* . https://angular.dev

Martin, R. C. (2018). *Clean Architecture: A Craftsman's Guide to Software Structure and Design* .

Prentice Hall.

Richardson, C. (2020). *Microservices Patterns: With examples in Java* . Manning Publications. Momjian, B. (2022). *PostgreSQL: Up and running* . O'Reilly Media.

García, M. (s. f.). *¿Qué es el Increment en Scrum?* ITtude. https://ittude-agile.com

SCRUMstudy™. (2017). *A Guide to the Scrum Body of Knowledge (SBOK Guide)* .

Scrum Guide. (2020). The Scrum Guide™ – The Definitive Guide to Scrum. Ken Schwaber & Jeff Sutherland.

Cohn, M. (2004). *User Stories Applied: For Agile Software Development* . Addison-Wesley. Comparasoftware. (2021). Planning Poker en Scrum.

Scrum.org. (2022). Agile Estimation Overview.

Nestrategia. (s. f.). *Back-End* . Recuperado de https://www.nestrategia.com

Anexos

Anexo 1: Scrip de la Base de Datos

Se hizo mención en el Capítulo IV en el apartado de Script de base de datos.

```
-- ======================================================

-- DATABASE: JASS_DB

-- PURPOSE: Gestión completa de facturación, recibos,

-- pagos, servicios especiales y deuda.

-- ======================================================
```

```
DROP TABLE IF EXISTS audit_log, receipt_details, receipts, receipts_correlative, payment_details, payments, consumer_special_services, special_services, consumers, consumer_activation_codes, users, roles CASCADE;

CREATE EXTENSION IF NOT EXISTS "uuid-ossp";

SET default_tablespace = '';

SET default_table_access_method = heap;

-- ========================================================

-- 1. ROLES -- Listo

-- ========================================================= CREATE TABLE roles (

id SERIAL PRIMARY KEY,

name VARCHAR(50) UNIQUE NOT NULL,

description TEXT,

active BOOLEAN DEFAULT TRUE,

created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP, updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP

);

CREATE OR REPLACE FUNCTION update_roles_timestamp() RETURNS TRIGGER AS $$

BEGIN

NEW.updated_at = NOW(); RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_update_roles_timestamp BEFORE UPDATE ON roles

FOR EACH ROW

EXECUTE FUNCTION update_roles_timestamp();

-- ========================================================

-- 2. USERS -- Refinada -- Pendiente

-- ========================================================= CREATE TABLE users (

id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),

username VARCHAR(100) UNIQUE NOT NULL,

password_hash TEXT NOT NULL,

role_id INT NOT NULL, consumer_id UUID,

active BOOLEAN DEFAULT TRUE,

created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
```

```
updated_at TIMESTAMP, created_by UUID,

FOREIGN KEY (role_id) REFERENCES roles(id), FOREIGN KEY (created_by) REFERENCES users(id)

);

-- FK solo después de crear consumers

-- (se agrega al final de consumers)

-- ====================================================

-- 3. CONSUMERS -- Listo

-- ====================================================

CREATE TABLE consumers (

id UUID PRIMARY KEY DEFAULT uuid_generate_v4(), dni VARCHAR(20) UNIQUE NOT NULL,

full_name VARCHAR(100) NOT NULL, address VARCHAR(200),

phone VARCHAR(20) UNIQUE,

total_debt NUMERIC(12,2) DEFAULT 0, status VARCHAR(20) DEFAULT 'ACTIVE',

created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP, email VARCHAR(150) UNIQUE

);

ALTER TABLE users

ADD CONSTRAINT fk_users_consumer

FOREIGN KEY (consumer_id) REFERENCES consumers(id);

-- ====================================================

-- 4. SPECIAL_SERVICES -- Listo

-- ====================================================

CREATE TABLE special_services ( id SERIAL PRIMARY KEY,

name VARCHAR(100) NOT NULL,

description TEXT,

cost NUMERIC(10,2) NOT NULL,

is_mandatory BOOLEAN DEFAULT FALSE

);

-- ====================================================

-- 5. CONSUMER_SPECIAL_SERVICES -- Listo

-- (solo registro de eventos, NO genera deuda directa)

-- ====================================================
```

```
CREATE TABLE consumer_special_services (

id UUID PRIMARY KEY DEFAULT uuid_generate_v4(), consumer_id UUID NOT NULL,

service_id INT NOT NULL,

-- Vigencia del servicio

start_date DATE NOT NULL DEFAULT CURRENT_DATE,

end_date DATE,

-- Estado del servicio para el consumidor

is_active BOOLEAN NOT NULL DEFAULT TRUE,

-- Auditoría básica

created_at TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,

updated_at TIMESTAMP,

-- Relaciones

CONSTRAINT fk_css_consumer FOREIGN KEY (consumer_id) REFERENCES consumers(id),

CONSTRAINT fk_css_service FOREIGN KEY (service_id) REFERENCES special_services(id)

);

-- ========================================================

-- REGLA DE NEGOCIO CLAVE

-- Un consumidor NO puede tener el mismo servicio activo

-- más de una vez, pero sí en el historial

-- ========================================================

CREATE UNIQUE INDEX ux_css_consumer_service_active ON consumer_special_services (consumer_id, service_id)
WHERE is_active = TRUE;

-- ========================================================

-- 6. PAYMENTS (facturas generadas) -- Refinado

-- ========================================================

CREATE TABLE payments (

id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),

consumer_id UUID NOT NULL,

period VARCHAR(7) NOT NULL, -- '2026-01'

payment_date DATE DEFAULT CURRENT_DATE, total_amount NUMERIC(12,2) NOT NULL, payment_method
VARCHAR(50),
```

```
status VARCHAR(20) NOT NULL DEFAULT 'PAID',

-- PAID | VOID

receipt_generated BOOLEAN DEFAULT FALSE, voided_at TIMESTAMP,

voided_by UUID, void_reason VARCHAR(255),

created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

FOREIGN KEY (consumer_id) REFERENCES consumers(id), FOREIGN KEY (voided_by) REFERENCES users(id)

);

-- ======================================================

-- 7. PAYMENT_DETAILS (detalles de cargos) -- Refinado

-- ======================================================

CREATE TABLE payment_details (

id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),

payment_id UUID NOT NULL, special_service_id INT NOT NULL, period VARCHAR(7) NOT NULL,

amount NUMERIC(10,2) NOT NULL,

status VARCHAR(20) NOT NULL DEFAULT 'PAID',

-- PAID | VOID

created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

FOREIGN KEY (payment_id) REFERENCES payments(id),

FOREIGN KEY (special_service_id) REFERENCES special_services(id)

);

-- ======================================================

-- 8. RECEIPTS -- Refinado -- Pendiente

-- ======================================================

CREATE TABLE receipts (

id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),

receipt_number VARCHAR(30) UNIQUE, consumer_id UUID NOT NULL,

total_amount NUMERIC(12,2) NOT NULL, payment_date DATE DEFAULT CURRENT_DATE, payment_method
VARCHAR(50),

status VARCHAR(20) NOT NULL DEFAULT 'ISSUED',

-- ISSUED | VOID

created_by UUID,
```

```
created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

voided_at TIMESTAMP, voided_by UUID, void_reason VARCHAR(255),

FOREIGN KEY (consumer_id) REFERENCES consumers(id), FOREIGN KEY (created_by) REFERENCES users(id), FOREIGN KEY (voided_by) REFERENCES users(id)

);

-- ======================================================

-- 9. RECEIPT_DETAILS -- Refinado -- Listo

-- ======================================================

CREATE TABLE receipt_details (

id UUID PRIMARY KEY DEFAULT uuid_generate_v4(),

receipt_id UUID NOT NULL, payment_id UUID NOT NULL, payment_detail_id UUID NOT NULL,

period VARCHAR(7) NOT NULL,

amount_paid NUMERIC(12,2) NOT NULL,

created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

FOREIGN KEY (receipt_id) REFERENCES receipts(id), FOREIGN KEY (payment_id) REFERENCES payments(id),

FOREIGN KEY (payment_detail_id) REFERENCES payment_details(id)

);

-- ======================================================

-- 10. RECEIPTS CORRELATIVE -- vive en la bd

-- ======================================================

CREATE TABLE receipts_correlative ( id SERIAL PRIMARY KEY,

year INT NOT NULL UNIQUE,

last_number INT DEFAULT 0 NOT NULL

);

CREATE OR REPLACE FUNCTION assign_receipt_number() RETURNS TRIGGER AS $$

DECLARE

v_year INT; v_next INT;

BEGIN

v_year := EXTRACT(YEAR FROM COALESCE(NEW.payment_date, CURRENT_DATE));

LOOP

UPDATE receipts_correlative
```

```
SET last_number = last_number + 1 WHERE year = v_year

RETURNING last_number INTO v_next;

IF FOUND THEN EXIT; END IF; BEGIN

INSERT INTO receipts_correlative (year, last_number) VALUES (v_year, 1)

RETURNING last_number INTO v_next; EXIT;

EXCEPTION WHEN unique_violation THEN END; END LOOP;

NEW.receipt_number := to_char(v_year,'FM0000') || '-' || lpad(v_next::text,6,'0'); RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_assign_receipt_number BEFORE INSERT ON receipts

FOR EACH ROW

EXECUTE FUNCTION assign_receipt_number();

-- ====================================================

-- 11. AUDIT LOG -- Pendiente

-- ====================================================

CREATE TABLE audit_log (

id UUID PRIMARY KEY DEFAULT uuid_generate_v4(), user_id UUID NOT NULL,

action VARCHAR(100) NOT NULL,

table_name VARCHAR(100), record_id TEXT NOT NULL,

created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

FOREIGN KEY (user_id) REFERENCES users(id)

);

-- ====================================================

-- 12. CORRECT DEBT CALCULATION

-- ====================================================

CREATE OR REPLACE FUNCTION update_total_debt() RETURNS TRIGGER AS $$

DECLARE

v_total_charges NUMERIC(12,2); v_total_paid NUMERIC(12,2);

BEGIN

-- Total cargos generados

SELECT COALESCE(SUM(pd.amount + pd.pending_balance), 0) INTO v_total_charges
```

```
FROM payment_details pd

JOIN payments p ON p.id = pd.payment_id WHERE p.consumer_id = NEW.consumer_id;

-- Total pagos realizados

SELECT COALESCE(SUM(rd.amount_paid), 0) INTO v_total_paid

FROM receipt_details rd

JOIN receipts r ON r.id = rd.receipt_id WHERE r.consumer_id = NEW.consumer_id;

UPDATE consumers

SET total_debt = v_total_charges - v_total_paid WHERE id = NEW.consumer_id;

RETURN NEW; END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER trg_update_debt_payment_details AFTER INSERT OR UPDATE ON payment_details

FOR EACH ROW EXECUTE FUNCTION update_total_debt();

CREATE TRIGGER trg_update_debt_payments AFTER INSERT OR UPDATE ON payments

FOR EACH ROW EXECUTE FUNCTION update_total_debt();

CREATE TRIGGER trg_update_debt_receipts AFTER INSERT OR UPDATE ON receipts

FOR EACH ROW EXECUTE FUNCTION update_total_debt();

-- ======================================================

-- 13. SEED ROLES

-- ======================================================

INSERT INTO roles (name, description) VALUES ('ADMIN', 'Full system access'),

('TREASURER', 'Handles payments'), ('SECRETARY', 'Manages consumers'), ('CONSUMER', 'Views personal account');

-- ======================================================

-- 14. INDEXES

-- ======================================================

CREATE INDEX idx_payments_consumer_id ON payments(consumer_id);

CREATE INDEX idx_payment_details_payment_id ON payment_details(payment_id); CREATE INDEX
idx_receipt_details_receipt_id ON receipt_details(receipt_id); CREATE INDEX idx_receipts_consumer_id ON
receipts(consumer_id);

-- ======================================================

-- 15. CONSUMER ACTIVATION CODES

-- ======================================================
```

```
CREATE TABLE consumer_activation_codes (

id UUID PRIMARY KEY DEFAULT uuid_generate_v4(), consumer_id UUID NOT NULL,

code VARCHAR(50) NOT NULL UNIQUE, used BOOLEAN DEFAULT FALSE,

created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

expires_at TIMESTAMP NOT NULL, used_at TIMESTAMP,

FOREIGN KEY (consumer_id) REFERENCES consumers(id)

);

-- ====================================================

-- Función para dar de baja códigos expirados

-- ====================================================

CREATE OR REPLACE FUNCTION desactivate_expired_codes() RETURNS VOID AS $$

BEGIN

UPDATE consumer_activation_codes SET used = TRUE

WHERE used = FALSE

AND expires_at <= CURRENT_TIMESTAMP; END;

$$ LANGUAGE plpgsql;
```

Anexo 2: Codificación Back-End

Se hizo mención en el Capítulo IV en el apartado de Codificación Back-End.

Figura 42

Clase Data-Initializer

Nota. Clase que se ejecuta cuando se inicia el back-end la primera vez, crea al usuario admin, con el cual se puede empezar a usar el sistema.