

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN

TRABAJO DE INTEGRACIÓN CURRICULAR PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN TECNOLOGÍAS DE LA INFORMACIÓN

TEMA:

” DESARROLLO DE SISTEMA MULTI MÓDULO DE ATENCIÓN EN SALA DE ESPERA CON INTERFAZ WEB Y PANEL DE CONTROL DE DESEMPEÑO. ”

Línea de investigación: Desarrollo, aplicación de software y cyber security (seguridad cibernética)



AUTOR: Dyland Ronaldo López Tierra
DIRECTOR: Mcs. Xavier Mauricio Rea Peñafiel

Ibarra – Ecuador

2026

UNIVERSIDAD TÉCNICA DEL NORTE
DIRECCIÓN DE BIBLIOTECA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO	
APELLIDOS Y NOMBRES:	DYLAND RONALDO LÓPEZ TIERRA

DATOS DE LA OBRA	
TÍTULO:	DESARROLLO DE SISTEMA MULTI MÓDULO DE ATENCIÓN EN SALA DE ESPERA CON INTERFAZ WEB Y PANEL DE CONTROL DE DESEMPEÑO.
AUTOR(ES):	DYLAND RONALDO LÓPEZ TIERRA
FECHA:	15/07/2026
CARRERA/PROGRAMA:	PREGRADO
TÍTULO POR EL QUE OPTA:	INGENIERO EN TECNOLOGÍAS DE LA INFORMACIÓN
DIRECTOR:	MCS. XAVIER MAURICIO REA PEÑAFIEL

2. CONSTANCIAS

El autor (es) manifiesta (n) que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es (son) el (los) titular (es) de los derechos patrimoniales, por lo que asume (n) la responsabilidad sobre el contenido de la misma y saldrá (n) en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 08 días del mes de 09 de 2026.

EL AUTOR:



(Firma): .....

Nombre: Dyland Ronaldo López Tierra

CERTIFICACIÓN DEL DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

15/07/2026

Mcs. Xavier Mauricio Rea Peñafiel
DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR

CERTIFICA:
Haber revisado el presente informe final del trabajo de Integración Curricular, el mismo que se ajusta a las normas vigentes de la Universidad Técnica del Norte; en consecuencia, autorizo su presentación para los fines legales pertinentes.

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
XAVIER MAURICIO REA PEÑAFIEL

Mcs. Xavier Mauricio Rea Peñafiel

C.C: 1002485744

APROBACIÓN DEL COMITÉ CALIFICADOR

El Comité Calificado del trabajo de Integración Curricular “ Desarrollo de Sistema Multi módulo de Atención en Sala de Espera con Interfaz Web y Panel de Control de Desempeño. ” elaborado por Dyland Ronaldo López Tierra , previo a la obtención del título del Ingeniero en Tecnologías de la Información , aprueba el presente informe de investigación en nombre de la Universidad Técnica del Norte:

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
XAVIER MAURICIO REA PEÑAFIEL

Mcs. Xavier Mauricio Rea Peñafiel

C.C: 1002485744

Validar únicamente en FirmaEC.
Firmado electrónicamente por:
VICTOR MANUEL CARANQUI SANCHEZ

Mcs. Victor Manuel Caranqui Sanchez

C.C: 1002006508

DEDICATORIA

Dedico este trabajo, en primer lugar, a Dios, por brindarme salud, fortaleza y sabiduría para superar cada desafío durante mi formación académica.

A mis padres Iván López e Irene Tierra, por su amor incondicional, esfuerzo y apoyo constante, quienes han sido el pilar fundamental para alcanzar esta meta.

A mis hermanos Brandon López, Carolina López y a toda mi familia, por su confianza, comprensión y palabras de aliento en los momentos más difíciles.

Finalmente, dedico este logro a todas las personas que, de una u otra manera, contribuyeron a mi crecimiento personal y profesional, motivándome a seguir adelante y a culminar con éxito esta etapa de mi vida.

Dyland Ronaldo López Tierra

AGRADECIMIENTO

Expreso mi más sincero agradecimiento a Dios por brindarme salud, fortaleza y sabiduría para culminar esta importante etapa de mi formación profesional.

Agradezco profundamente a mis padres y a mi familia por su amor, apoyo incondicional, comprensión y motivación constante durante todo este proceso. Su confianza y esfuerzo han sido fundamentales para alcanzar esta meta.

Mi agradecimiento a la Universidad Técnica del Norte y a los docentes de la carrera de Tecnologías de la Información, quienes compartieron sus conocimientos, experiencias y orientación, contribuyendo significativamente a mi formación académica y profesional.

De manera especial, agradezco a mi director de tesis por su guía, asesoramiento y valiosas observaciones durante el desarrollo de esta investigación, las cuales permitieron fortalecer la calidad del presente trabajo.

Asimismo, expreso mi gratitud a la Empresa Pública de Movilidad del Norte "MOVIDELNOR" por brindar la apertura y las facilidades necesarias para el desarrollo del proyecto, permitiendo la recopilación de información y la implementación de la propuesta tecnológica.

Finalmente, agradezco a todas las personas que, de una u otra forma, contribuyeron con su apoyo, consejos y colaboración para hacer posible la culminación de este trabajo de investigación.

Dyland Ronaldo López Tierra

ÍNDICE DE CONTENIDOS

DEDICATORIA 6

AGRADECIMIENTO 7

ÍNDICE DE FIGURAS 13

ÍNDICE DE TABLAS 13

RESUMEN 15

ABSTRACT 16

CAPÍTULO I

1.1 Planteamiento del problema 2

1.2 Objetivos 3

1.2.1 Objetivo General 3

1.2.2 Objetivos Específicos 3

1.3 Alcance 4

1.4 Justificación 4

1.4.1 Ámbito técnico y de ingeniería 5

1.4.2 Metodologías Ágiles 5

1.4.3 Enfoque en Atributos de Calidad 6

CAPÍTULO II

2.1 Antecedentes 7

2.1.1 Simulación de sistemas de colas para la optimización del servicio al cliente en el sector de restaurantes. 7

2.1.2 Sistema de gestión de colas de una sala de espera física 7

2.1.3 Modelo de teoría de colas y simulación para la asignación del recurso humano en un sistema operativo de servicios 8

2.1.4 Mejora del servicio de atención aplicando teoría de colas para reducir tiempos de espera en el centro médico P&P Anatomy 8

2.1.5 Mejora en la atención al cliente a través de la teoría de colas: Caso Distribuidora El Hueco S.A.S 9

2.2 Bases teóricas 9

2.2.1 Sistemas de información 9

2.2.2 Sistemas de gestión de colas 10

2.2.3 Ingeniería de software 10

2.3 Usabilidad y Experiencia de Usuario (UX) en sistemas de atención 11

2.3.1 Concepto de usabilidad en sistemas informáticos 11
2.3.2 Experiencia de Usuario (UX): definición y evolución 11
2.3.3 UX en sistemas de atención al cliente 12
2.3.4 Sistemas de colas y percepción del tiempo de espera 12
2.3.5 Diseño centrado en el usuario (User-Centered Design) 12
2.3.6 Interfaces web para salas de espera 13
2.3.7 Accesibilidad web en sistemas de atención 13
2.3.8 Usabilidad en aplicaciones web desarrolladas con Flask 14
2.3.9 Relación entre UX y eficiencia operativa 14
2.3.10 Usabilidad según la norma ISO/IEC 25010 14
2.4 Arquitecturas web multimódulo 15
2.5 Flask 15

2.5.1 Qué es Flask 15
2.5.2 Cuál es la arquitectura 16
2.5.3 Tabla de beneficios y desventajas de usar Flask 16
2.5.4 Qué es el ruteador de Flask 16
2.5.5 Qué son las plantillas en Flask 17
2.5.6 Qué son los API REST 18
2.5.7 Acceso a bases de datos desde Flask 18
CAPÍTULO III
3.1 Enfoque y Tipo de Investigación 19
3.1.1 Enfoque de investigación: Cuantitativo 19
3.1.2 Tipo de investigación: Aplicada – Desarrollo tecnológico 19
3.2 Diseño de la Investigación 20
3.2.1 Diseño metodológico general: Proceso de desarrollo de software 20
3.2.2 Fases del diseño: Análisis, diseño, implementación, pruebas y validación 20
3.3 Metodología de Desarrollo de Software 21
3.3.1 Selección del modelo de desarrollo: Metodología ágil (Scrum) 21
3.4 Técnicas e Instrumentos para la Validación 21
CAPÍTULO IV
4.1 Product Backlog 22
4.2 Planificación de Sprints 24
4.3 Diseño de la base de datos 24
4.4 Diagrama de casos de uso 25
4.5 Desarrollo de los sprints 26

4.6 Pruebas 34
4.6.1 Pruebas funcionales 34
4.6.2 Pruebas de rendimiento 35
4.6.3 Pruebas de usabilidad (SUS) 35
4.7 Validación 36
4.7.1 Validación de historias de usuario 36
4.7.2 Validación con usuarios reales 37
4.7.3 Validación técnica 37
4.7.4 Conclusiones de validación 38
CONCLUSIONES 38
RECOMENDACIONES 41
Recomendaciones 41
BIBLIOGRAFÍA 42

ÍNDICE DE FIGURAS

Figura 1 Arquitectura de framework modelo MVT 16

Figura 2 Diseño de Base de Datos 25
Figura 3 Diagrama de casos de uso 26
Figura 4 Estructura de carpetas 27
Figura 5 Pantalla de login 28
Figura 6 Pantalla de funcionario común 28
Figura 7 Pantalla de Súper funcionario 29
Figura 8 Pantalla publica de visualización de turnos 30
Figura 9 Pantalla publica pasa solicitar turnos segun el modulo necesario 30
Figura 10 Pantalla de funcionario con cronometro y turnos del modulo en cola . . 31
Figura 11 Pantalla de transferencias de turnos a diferentes módulos 32
Figura 12 Pantalla de medición de rendimiento 32
Figura 13 Exportación de métricas en formato CSV 33
Figura 14 Exportación de métricas en formato PDF 33

ÍNDICE DE TABLAS

Tabla 1 Ventajas y desventajas del framework Flask 17

Tabla 2 Product Backlog - Historias de usuario 23
Tabla 3 Planificación de Sprints 24
Tabla 4 Resultados de validación por historia de usuario 36

RESUMEN

La atención al ciudadano en instituciones públicas presenta problemas relacionados con los tiempos de espera, la organización de los procesos de servicio y la limitada disponibilidad de herramientas tecnológicas que permitan optimizar la gestión de usuarios. Con el fin de contribuir a la mejora de estos procesos, esta investigación desarrolló un sistema multimódulo de atención en sala de espera con interfaz web y panel de control de desempeño para la Empresa Pública de Movilidad del Norte ”MOVIDELNOR”, utilizando la metodología Scrum como marco de trabajo para planificar, desarrollar y validar las funcionalidades del sistema de manera iterativa. Asimismo, la solución fue implementada empleando Flask como framework de desarrollo y MariaDB como sistema gestor de bases de datos, incorporando módulos para la generación de turnos, atención de usuarios, visualización pública en tiempo real y monitoreo de indicadores de desempeño mediante un panel administrativo. Además, se integraron mecanismos para el registro de atenciones, control de tiempos y generación de reportes, permitiendo una supervisión continua de la operación. Los resultados evidenciaron un funcionamiento eficiente del sistema, alcanzando una disponibilidad del noventa y nueve por ciento, y una puntuación de usabilidad de 85/100 en la escala SUS, obtenida mediante encuestas aplicadas a los funcionarios, clasificación considerada buena según dicha escala. De igual manera, las pruebas funcionales demostraron un desempeño estable y una respuesta adecuada durante la atención de usuarios. En consecuencia, se concluye que la implementación del sistema mejora la organización de la atención, optimiza la gestión de turnos, facilita el seguimiento de los procesos, fortalece la experiencia tanto de funcionarios como de ciudadanos y contribuye significativamente a incrementar la eficiencia operativa, la transparencia, la calidad del servicio ofrecido y la satisfacción institucional general.

Palabras clave: SCRUM , PYTHON , FLASK

ABSTRACT

Citizen service in public institutions faces challenges regarding wait times, the organization of service workflows, and the limited availability of technological tools to optimize user management. To help improve these processes, this research developed a multi-module waiting-room service system for ”MOVIDELNOR”featuring a web interface and a performance dashboard—using the Scrum methodology as a framework to plan, develop, and validate system functionalities iteratively. The solution was implemented using the Flask development framework and MariaDB as the database management system, incorporating modules for queue management, user service, real-time public display, and performance monitoring via an administrative dashboard. Additionally, mechanisms were integrated to log service interactions, track time, and generate reports, enabling continuous operational oversight. Results demonstrated efficient system performance, achieving ninety-nine percent availability and a usability score of 85/100 on the System Usability Scale based on staff surveys a rating considered good on that scale. Functional testing also confirmed stable performance and adequate responsiveness during user service. Consequently, the study concludes that implementing this system improves service organization, optimizes queue management, facilitates process tracking, enhances the experience for both staff and citizens, and significantly contributes to increased operational efficiency, transparency, and service quality within the institution.

Resumen

Keywords: SCRUM , PYTHON , FLASK

CAPÍTULO I

INTRODUCCIÓN

1.1 Planteamiento del problema

En los centros y organismos de la Empresa Pública de Movilidad del Norte "MOVIDEL-NOR" que tienen servicio de atención al público, uno de los mayores desafíos consiste en el eficaz manejo de las salas de espera. La carencia de sistemas centralizados genera desorganización para los tiempos de espera incrementados, falta de información en tiempo medio y truncamiento para medir el desempeño de los funcionarios. Esto genera el malestar de los usuarios, disminución de eficiencia productiva y limitaciones para tomar decisiones estratégicas por falta de información consolidada ni reportes transparentes que permitan conocer zonas de oportunidad [1].

Las mayores causas del problema radican en la falta de herramientas tecnológicas que centralicen los procesos de atención, con una interfaz amigable que facilite la atención de los operadores hacia los usuarios y la falta de información procesada que permita brindar indicadores para toma de decisiones. A lo anterior se agrega que las diferentes herramientas clásicas tampoco tienen escalabilidad y garantías de seguridad para responder necesidades actuales, además de que no permiten la integración entre los diferentes módulos de atención, administración y diagnóstico[2].

Las secuelas son variadas: los usuarios perciben una deficiente experiencia al no tener información clara sobre el turno y/o tiempo de espera estimado; los funcionarios tienen una carga de trabajo desorganizada que genera un mal desempeño [3]; los administradores no tienen métricas fiables para comparar el eficaz servicio para permitir mejorar el trabajo continuo.

A partir la aplicación de la ingeniería de software, el problema se resolverá con el estudio de los requerimientos, con el diseño estándar con fundamento sobre las últimas arquitecturas, con trabajo sobre interfaces accesibles y responsivas, así como con implementar formas de panel de control que permitan aglomerar métricas en tiempo real y automatizaciones de reportes. Se busca así aumentar el flujo de atención óptima, dar una experiencia óptima al usuario y le dará a la compañía herramienta sólida para el diseño de decisiones estratégicas. Ante esta situación, nace la necesidad de diseñar e implementar un Sistema Multi-módulo de atención en sala de espera con interfaz web y panel de control de desempeño[3].

1.2 Objetivos

1.2.1 Objetivo General

Desarrollar una plataforma web multi módulo para la gestión de atención en sala de espera, con interfaz adaptable y panel de control gráfico, que optimice el flujo de atención y mejore la experiencia de usuarios y administradores.

1.2.2 Objetivos Específicos

- Definir y analizar los requerimientos funcionales y no funcionales del sistema, considerando aspectos de seguridad, escalabilidad, accesibilidad y usabilidad, con el fin de establecer una arquitectura web adecuada y sostenible.
- Implementar los módulos funcionales de la plataforma web, integrando la gestión de turnos, el registro y seguimiento de usuarios, y la administración del servicio, junto con componentes de visualización de datos y generación de reportes que faciliten el control operativo y estratégico del sistema.
- Evaluar la usabilidad de la plataforma web mediante pruebas con usuarios y métricas estandarizadas, con el fin de validar su facilidad de uso, accesibilidad y satisfacción del

usuario en la optimización del flujo de atención.

1.3 Alcance

El trabajo en esta propuesta consiste en el diseño e implementación de un sistema multi módulo de atención en sala de espera con panel de control de desempeño e interfaz web, realizando pruebas técnicas y de funcionamiento de la solución.

- El levantamiento y diseño de requisitos funcionales y no funcionales.
- El diseño de una interfaz fácil de usar que permita que los usuarios y los funcionarios se comuniquen.
- La implementación de una interfaz de panel de control con métricas y reportes automatizados.
- La verificación del sistema en escenarios de prueba que imitarán un entorno de sala de espera institucional.

No se considera la etapa de lanzamiento a producción del sistema en gran medida ni tampoco su interoperabilidad con plataformas externas. Se considera el compromiso de entrega funcional que corresponda al cumplimientos de los objetivos específicos y que se tenga una sólida base para fases posteriores de implementación.

1.4 Justificación

Este trabajo está orientado a la elaboración de un Sistema multi módulo de atención en la sala de espera con interfaz web y un panel de control de desempeño, y su pertinencia puede ser vista desde varios ángulos. Se trata de una situación problemática en los servicios de atención al ciudadano en la Empresa Pública de Movilidad del Norte “MOVIDELNOR”.

La carencia de este tipo de sistemas no solo se traduce en un déficit tecnológico frente a las expectativas actuales, sino también en una baja eficiencia operativa, una percepción negativa por parte del ciudadano y limitaciones en la gestión adecuada de los procesos de atención. En este contexto, resulta necesario analizar la pertinencia de soluciones tecnológicas que permitan optimizar el servicio y apoyar la toma de decisiones [4]. A continuación, se expone la justificación en los diferentes ámbitos.

1.4.1 Ámbito técnico y de ingeniería

Desde un punto de vista de la ingeniería de software, el proyecto constituye un ejemplo importante de sus aportaciones prácticas aplicando principios de diseño de interés para la solución de un problema real [5].

Se logrará obtener un artefacto de software concreto que dará fe de la aplicabilidad de una arquitectura multi módulo centralizada. El Valor técnico radica en la arquitectura de referencia: el diseño y la implementación se etiquetarán como un caso práctico para el desarrollo de sistemas similares en el ámbito público, dedicándonos especialmente a cómo se pueden integrar los diferentes módulos de gestión de colas, las interfaces de usuario y los paneles de control analítico dentro de una única plataforma [2].

1.4.2 Metodologías Ágiles

El presente proyecto nos dará reflexiones asimiladas acerca de la aplicación de ciclos de vida de desarrollo Scrum , gestión de requisitos, diseño de interfaces de usuarios (UI/UX) para los distintos actores, implementación de paneles de control de visualización de los datos procesados [6].

1.4.3 Enfoque en Atributos de Calidad

No solo se incentiva el hecho de ofrecer funcionalidad, sino de construir un sistema que refuerce la escalabilidad (para futuras ampliaciones), la usabilidad (para realizar una adopción de usuarios efectiva), la seguridad (en la manipulación de datos de atención a las personas) y la mantenibilidad (código bien definido y documentado)**FernndezJarrn2024**.

CAPÍTULO II

MARCO TEÓRICO

2.1 Antecedentes

2.1.1 Simulación de sistemas de colas para la optimización del servicio al cliente en el sector de restaurantes

Esta revisión sistemática se ha centrado en el análisis de la simulación de sistemas de colas como vía de mejora del servicio al cliente en los restaurantes. Se identifican las largas colas y los tiempos de espera, especialmente en establecimientos de comida rápida, como un problema habitual que genera insatisfacción en los clientes.

La revisión sistemática, desarrollada bajo la metodología PRISMA, evidencia que la simulación de sistemas de colas aporta un valor significativo a los gerentes de restaurantes, ya que permite comprender el comportamiento real de las colas y tomar decisiones informadas sobre la capacidad de los recursos, como el número de servidores. Como resultado, se logra una mejora sustancial en la calidad del servicio al cliente [7].

2.1.2 Sistema de gestión de colas de una sala de espera física

Esta investigación aborda la problemática de las largas colas y la incomodidad generada en la sala de espera de la oficina de vivienda y hábitat. El objetivo principal fue desarrollar una aplicación web que permita gestionar la cola de atención, con el fin de mejorar el servicio al cliente y la organización interna.

La solución propuesta consistió en una aplicación web desarrollada utilizando el framework

Laravel versión 8.0, bajo el patrón Modelo–Vista–Controlador (MVC), y una base de datos MySQL. El sistema facilita la gestión de la sala de espera, optimiza el uso del tiempo y permite replicar la solución de manera sencilla en otras instituciones que deseen automatizar la administración de sus colas físicas [8].

2.1.3 Modelo de teoría de colas y simulación para la asignación del recurso humano en un sistema operativo de servicios

Este estudio de campo demostró que la asignación del recurso humano en los sistemas de atención no debe ser estática. A partir del análisis de la teoría de colas y la simulación, se evidenció que los gerentes pueden implementar una gestión dinámica de los servidores en los procesos de atención al cliente.

Como resultado, el modelo de teoría de colas se transforma en un sistema de control operacional continuo, logrando que los tiempos de espera en fila se mantengan consistentemente por debajo de los 20 minutos. Además, se obtuvo un ahorro del 16.36 % en horas de trabajo, lo que refleja una mejora tanto en la eficiencia operativa como en la calidad del servicio [9].

2.1.4 Mejora del servicio de atención aplicando teoría de colas para reducir tiempos de espera en el centro médico P&P Anatomy

El objetivo principal de esta investigación fue proponer mejoras en el servicio de atención mediante la aplicación de la teoría de colas, con el fin de reducir los tiempos de espera en el centro médico P&P Anatomy, ubicado en la ciudad de Chiclayo.

Se aplicó el modelo de colas M/M/s y se realizó una simulación utilizando el software ProModel para determinar la cantidad óptima de especialistas necesarios para la atención. Los resultados evidenciaron que la propuesta es viable, logrando una reducción de los tiempos de espera del 30 %. Asimismo, el análisis costo–beneficio arrojó un indicador B/C de S/. 1,10, lo que confirma la rentabilidad del proyecto [10].

2.1.5 Mejora en la atención al cliente a través de la teoría de colas: Caso Distribuidora El Hueco S.A.S

La empresa Distribuidora El Hueco S.A.S, dedicada a la comercialización de productos, enfrentaba serios problemas debido a las largas filas de espera en las cajas durante la temporada alta, con un flujo diario estimado entre 1000 y 1200 clientes. Esta situación generaba insatisfacción y pérdida de clientes.

El objetivo del estudio fue mejorar el proceso de atención al cliente mediante la implementación de la teoría de colas, optimizando los indicadores de desempeño y la satisfacción del cliente. Los resultados mostraron una mejora significativa en dichos indicadores, reflejándose en un aumento progresivo de la satisfacción de los clientes y en un mayor volumen de personas atendidas en el establecimiento [11].

2.2 Bases teóricas

2.2.1 Sistemas de información

Un sistema de información se define como un conjunto organizado de componentes que interactúan entre sí con el propósito de recopilar, procesar, almacenar y distribuir información para apoyar la toma de decisiones, la coordinación y el control dentro de una organización. En el contexto institucional, los sistemas de información permiten automatizar procesos, reducir errores humanos y mejorar la eficiencia operativa[12].

Los sistemas de información modernos suelen apoyarse en tecnologías web, lo que facilita el acceso remoto, la escalabilidad y la integración con otros sistemas. En particular, los sistemas orientados a la atención al público requieren un diseño centrado en el usuario, que garantice claridad, rapidez y confiabilidad en la gestión de la información[13].

2.2.2 Sistemas de gestión de colas

La teoría de colas constituye una disciplina matemática que estudia el comportamiento de líneas de espera y los procesos de servicio asociados. Su aplicación en sistemas de atención permite modelar escenarios reales en los que los usuarios llegan de manera aleatoria y son atendidos por uno o varios servidores[14].

En los sistemas de gestión de colas, la automatización del proceso de asignación de turnos contribuye a reducir la desorganización, mejorar la percepción de justicia en la atención y proporcionar información clara sobre los tiempos estimados de espera. Cuando estos sistemas se implementan mediante plataformas web, se potencia su capacidad de adaptación a distintos contextos institucionales[15].

2.2.3 Ingeniería de software

La ingeniería de software es la disciplina que aplica principios científicos, métodos y herramientas para el desarrollo, operación y mantenimiento de sistemas de software de alta calidad. Su objetivo principal es producir soluciones confiables, eficientes y alineadas con las necesidades del usuario y de la organización.

En proyectos de desarrollo de sistemas web, la ingeniería de software abarca actividades como el levantamiento de requisitos, el diseño de la arquitectura, la implementación de funcionalidades, las pruebas y la documentación. La correcta aplicación de estos principios resulta fundamental para garantizar la sostenibilidad y mantenibilidad del sistema a largo plazo[16].

2.3 Usabilidad y Experiencia de Usuario (UX) en sistemas de atención

2.3.1 Concepto de usabilidad en sistemas informáticos

La usabilidad constituye un atributo fundamental de calidad del software, especialmente en sistemas orientados a la atención al público. Se refiere al grado en que un producto de software puede ser utilizado por usuarios específicos para lograr objetivos definidos con eficacia, eficiencia y satisfacción dentro de un contexto de uso determinado. En entornos institucionales, una alta usabilidad reduce errores operativos, tiempos de aprendizaje y resistencia al uso del sistema, aspectos críticos en escenarios de atención ciudadana**iso25010_2023**.

En los sistemas de colas y atención al cliente, la usabilidad se relaciona directamente con la claridad de la interfaz, la facilidad para comprender el estado del turno y la simplicidad de las acciones que el usuario debe realizar. Un diseño poco usable incrementa la percepción de espera, genera frustración y deteriora la imagen institucional.

2.3.2 Experiencia de Usuario (UX): definición y evolución

La Experiencia de Usuario (UX) amplía el concepto tradicional de usabilidad al incorporar factores emocionales, cognitivos y contextuales que influyen en la interacción entre el usuario y el sistema. La Experiencia de Usuario (UX) no se limita al uso funcional del software, sino que considera cómo se siente el usuario antes, durante y después de la interacción.

La evolución del enfoque UX ha llevado a que los sistemas modernos prioricen la empatía con el usuario, la consistencia visual, la retroalimentación inmediata y la reducción de la incertidumbre. En sistemas de atención en salas de espera, estos elementos resultan determinantes para mejorar la percepción del servicio, incluso cuando los tiempos de espera no pueden reducirse significativamente [17], [18].

2.3.3 UX en sistemas de atención al cliente

Los sistemas de atención al cliente presentan particularidades que los diferencian de otros tipos de software. Los usuarios suelen encontrarse en situaciones de urgencia, estrés o expectativa, lo que incrementa la sensibilidad frente a fallos de comunicación o interfaces confusas. Por esta razón, la UX en estos sistemas debe enfocarse en la transparencia del proceso y la previsibilidad del servicio.

Un sistema de atención bien diseñado informa claramente al usuario sobre su posición en la cola, el tiempo estimado de espera y los pasos siguientes del proceso. Esta información reduce la ansiedad y mejora la percepción de control del usuario sobre la situación[3].

2.3.4 Sistemas de colas y percepción del tiempo de espera

Diversos estudios han demostrado que la percepción del tiempo de espera no depende únicamente de su duración real, sino también de la calidad de la información proporcionada al usuario durante la espera. La ausencia de información genera la sensación de desorden y prolongación del tiempo, mientras que interfaces informativas y visuales reducen la percepción negativa de la espera[15].

En este contexto, los sistemas de colas informatizados desempeñan un papel clave al proporcionar datos en tiempo real, mensajes de estado y señales visuales que mantienen informado al usuario. La UX se convierte así en un componente estratégico para la gestión de la espera.

2.3.5 Diseño centrado en el usuario (User-Centered Design)

El diseño centrado en el usuario es un enfoque metodológico que sitúa al usuario final como eje principal del proceso de diseño y desarrollo del software. Este enfoque implica comprender las necesidades, capacidades y limitaciones de los usuarios desde las primeras etapas del proyecto.

En sistemas web de atención, el diseño centrado en el usuario permite crear interfaces intuitivas, reducir la carga cognitiva y garantizar que las funcionalidades respondan a escenarios reales de uso. La aplicación de este enfoque resulta coherente con metodologías ágiles como Scrum, donde la retroalimentación continua del usuario es fundamental [19].

2.3.6 Interfaces web para salas de espera

Las interfaces web utilizadas en salas de espera deben caracterizarse por su simplicidad, claridad visual y capacidad de adaptación a distintos dispositivos. Elementos como tipografías legibles, contrastes adecuados y jerarquía visual clara facilitan la comprensión de la información presentada.

En el caso de plataformas desarrolladas con Flask, el diseño de la interfaz web se apoya en la correcta separación entre la lógica del negocio y la presentación, permitiendo la creación de vistas limpias y consistentes que favorecen la experiencia del usuario [20].

2.3.7 Accesibilidad web en sistemas de atención

La accesibilidad web es un componente esencial de la experiencia de Usuario (UX), especialmente en instituciones públicas que atienden a una población diversa. Un sistema accesible garantiza que personas con discapacidades visuales, auditivas o motrices puedan interactuar con la plataforma sin barreras [21].

La incorporación de principios de accesibilidad en el diseño de sistemas de atención no solo cumple con normativas internacionales, sino que también amplía el alcance del servicio y mejora la percepción de inclusión y responsabilidad social de la institución.

2.3.8 Usabilidad en aplicaciones web desarrolladas con Flask

Flask, como framework web ligero en Python, permite desarrollar aplicaciones flexibles y modulares. Esta característica facilita la implementación de principios de usabilidad, al permitir adaptar la interfaz y las funcionalidades según las necesidades del usuario.

La correcta estructuración de rutas, vistas y plantillas contribuye a una navegación coherente y predecible, elementos fundamentales para garantizar una experiencia de usuario positiva en sistemas de atención [16].

2.3.9 Relación entre UX y eficiencia operativa

Una buena experiencia de usuario no solo beneficia al ciudadano, sino también a la institución. Sistemas con alta usabilidad reducen errores, consultas repetitivas y tiempos improductivos, lo que se traduce en una mayor eficiencia operativa.

En sistemas de colas, una UX adecuada contribuye a un flujo de atención más ordenado, mejor distribución de la carga de trabajo y mayor control por parte de los administradores [22].

2.3.10 Usabilidad según la norma ISO/IEC 25010

La norma ISO/IEC 25010 define la usabilidad como una característica de calidad del software, compuesta por subcaracterísticas como la adecuación reconocible, la aprendibilidad, la operabilidad, la protección contra errores de usuario y la estética de la interfaz **iso25010_2023**.

En el presente proyecto, la usabilidad constituye un criterio clave de validación del sistema, ya que su correcta implementación impacta directamente en la experiencia del usuario y en la efectividad del proceso de atención.

2.4 Arquitecturas web multimódulo

Las arquitecturas web multi módulo se caracterizan por dividir un sistema en componentes funcionales independientes pero interconectados, cada uno encargado de cumplir una función específica. Este enfoque favorece la escalabilidad, la mantenibilidad y la reutilización del software [13].

En un sistema multi módulo de atención en sala de espera, los módulos pueden incluir la gestión de turnos, el registro de usuarios, la administración del servicio y el análisis de desempeño mediante paneles de control. La comunicación entre módulos se realiza a través de interfaces bien definidas, lo que permite la integración y evolución progresiva del sistema.

El uso de arquitecturas modernas, apoyadas en tecnologías web y principios de diseño modular, facilita la adaptación del sistema a nuevas necesidades institucionales, así como la incorporación de mejoras futuras sin afectar el funcionamiento global. De esta manera, las arquitecturas web multi módulo constituyen una base sólida para el desarrollo de sistemas de atención eficientes y orientados a la toma de decisiones basada en datos [23].

2.5 Flask

2.5.1 Qué es Flask

Flask es un microframework ligero y flexible que es utilizado para el desarrollo web en conjunto con Python, tiene herramientas esenciales para tener un flujo de desarrollo web ágil y escalable.

Esto ayuda a los desarrolladores a utilizar bibliotecas externas para bases de datos. siendo perfecto para prototipos, APIs y micro servicios, a diferencia de frameworks más grandes como Django[24].

2.5.2 Cuál es la arquitectura

La arquitectura de Flask se denomina como MVT(Model-View-Template) este modelo contiene tres capas principales las cuales son comúnmente utilizadas en la creación de aplicaciones web. Como se puede observar en la **Figura 1**

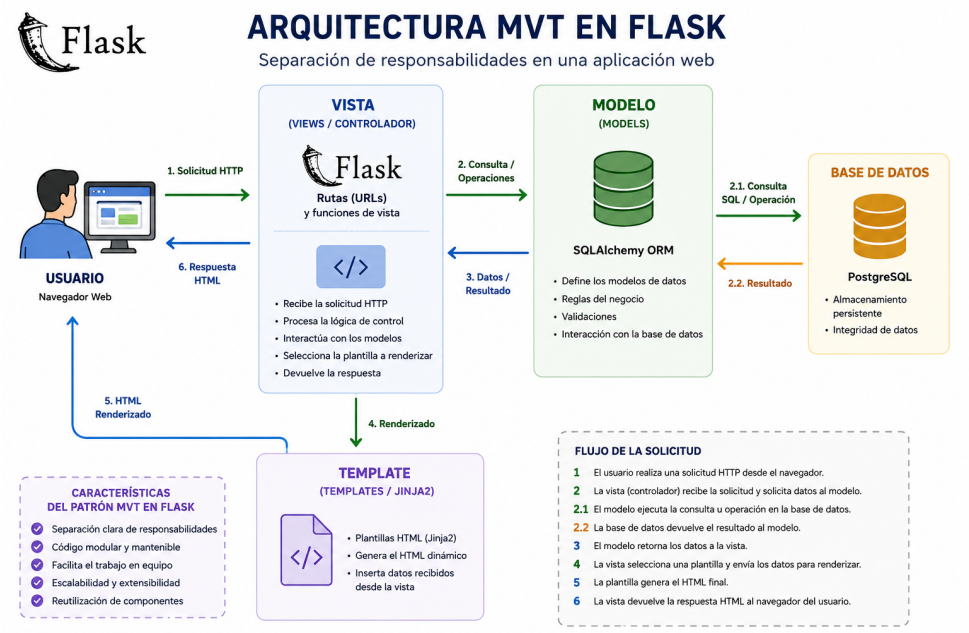


Figura 1.
Arquitectura de framework modelo MVT (Adaptado de [25])

La capa Model que es responsable de la lógica y la persistencia del dominio. La capa View interactúa con el modelo y ejecutando solicitudes a servicios externos. La capa Template se encarga de la presentación de la interfaz de usuario[25].

2.5.3 Tabla de beneficios y desventajas de usar Flask

2.5.4 Qué es el ruteador de Flask

El ruteador de Flask es el mecanismo que enlaza URLs con funciones en aplicaciones es parte del núcleo del framework, esto ayuda a tener un mayor dinamismo y prevenir errores en el cambio de estructuras en los enlaces. este también permite filtrar peticiones por métodos HTTP

Categoría	Beneficios (Ventajas)	Desventajas
Arquitectura	**Ligero y minimalista:** Solo incluye lo esencial, lo que reduce el consumo de memoria y acelera el inicio.	**Falta de funciones integradas:** No incluye ORM ni sistema de autenticación nativo (requiere extensiones manuales).
Desarrollo	**Flexibilidad total:** Libertad para elegir bases de datos, librerías y estructura del proyecto.	**Sobrecarga de decisiones:** El desarrollador debe investigar y elegir cada componente externo.
Aprendizaje	**Curva de aprendizaje suave:** Es ideal para principiantes y fácil de entender si ya conoces Python.	**Riesgo de mal código:** Al ser menos restrictivo, es más fácil para desarrolladores novatos escribir código desordenado.
Rendimiento	**Mayor velocidad:** Menos capas de abstracción permiten un rendimiento ligeramente superior a frameworks pesados.	**Escalabilidad compleja:** En proyectos masivos, gestionar manualmente las dependencias puede volverse ineficiente.
Operaciones	**Fácil de desplegar:** Su base de código pequeña facilita el despliegue en producción y prototipado rápido.	**Costos de mantenimiento:** Si una extensión de terceros queda obsoleta, el desarrollador debe buscar reemplazos rápidamente.

Tabla 1.
Ventajas y desventajas del framework Flask

como: GET, POST, PUT o DELETE[24].

2.5.5 Qué son las plantillas en Flask

En Flask, las plantillas (templates) son archivos (HTML) dinámicos que permiten generar páginas web unificando una estructura visual (HTML) con datos del backend con Python. Flask utiliza el motor de plantillas Jinja2. lo cual facilita en gran medida el desarrollo de aplicaciones. Flask procesa la lógica de Python para enviar datos a la plantilla en la cual Jinja2 reemplaza variables y ejecuta estructuras, el navegador recibe (HTML) reemplazado[24].

2.5.6 Qué son los API REST

Las Api REST en Flask se implementa como rutas que reciben solicitudes HTTP GET,POST,PUT, DELETE en formato JSON en lugar de paginas HTML esto también ayudar a las conexiones con servicios externos[25].

2.5.7 Acceso a bases de datos desde Flask

El acceso a bases de datos desde Flask consiste en la consulta y manipulación de datos almacenados en un sistema gestor de bases de datos (SGBD) desde una aplicación web o API desarrollada con Flask. Esto permite mantener información, gestionar usuarios, registrar transacciones y mantener la integridad de los datos. Para esto se utiliza una herramienta externa como SQLAlchemy el cual es un potente mapeador objeto relacional o ORM el cual permite interactuar con la base de datos usando clases de Python en lugar de consultas SQL crudas[26].

CAPÍTULO III

MATERIALES Y MÉTODOS

3.1 Enfoque y Tipo de Investigación

3.1.1 Enfoque de investigación: Cuantitativo

El presente trabajo adopta un enfoque de investigación cuantitativo, el cual, desde la perspectiva de la ingeniería de software, permite medir fenómenos y analizar resultados mediante técnicas estadísticas asociadas a métricas de desempeño, tiempos de respuesta y niveles de satisfacción de los usuarios [27]. En el contexto del desarrollo de sistemas, este enfoque resulta adecuado para la recolección y análisis de datos numéricos que puedan ser cuantificados y sometidos a análisis estadístico.

3.1.2 Tipo de investigación: Aplicada – Desarrollo tecnológico

La investigación se clasifica como aplicada de desarrollo tecnológico, ya que su propósito fundamental es resolver un problema práctico específico mediante la creación de un producto tecnológico. El desarrollo de un sistema multi módulo de atención en sala de espera constituye un artefacto tecnológico cuya creación, implementación y validación responden a la necesidad de optimizar los procesos de atención al ciudadano, en concordancia con los lineamientos establecidos para el desarrollo e implementación de software bajo buenas prácticas [5].

3.2 Diseño de la Investigación

3.2.1 Diseño metodológico general: Proceso de desarrollo de software

El diseño metodológico general del proyecto se estructura en torno a un proceso sistemático de desarrollo de software. La ingeniería de software se define como la disciplina que aplica principios científicos, métodos y herramientas para el desarrollo de sistemas de software de alta calidad, alineados con las necesidades del usuario y de la organización [27]. Bajo esta perspectiva, el diseño de investigación contempla cada una de las etapas necesarias para transformar los requisitos funcionales y no funcionales en un producto de software funcional, fiable y mantenible.

3.2.2 Fases del diseño: Análisis, diseño, implementación, pruebas y validación

El proceso de desarrollo se organiza en cinco fases fundamentales que constituyen el ciclo de vida del software: análisis, diseño, implementación, pruebas y validación. En la fase de análisis se realiza el levantamiento y especificación de requisitos funcionales y no funcionales del sistema. En la fase de diseño se define la arquitectura multi módulo, las interfaces de usuario y la estructura de datos subyacente. La implementación corresponde a la codificación del sistema utilizando el framework Flask y las tecnologías asociadas. Las pruebas comprenden la verificación del correcto funcionamiento de cada módulo y la integración entre ellos. Finalmente, la validación implica la evaluación del sistema frente a los criterios de usabilidad, eficiencia y satisfacción del usuario [27].

3.3 Metodología de Desarrollo de Software

3.3.1 Selección del modelo de desarrollo: Metodología ágil (Scrum)

Para el desarrollo del presente proyecto se ha seleccionado la metodología ágil Scrum. Diversos estudios destacan que las metodologías ágiles presentan ventajas que permiten una gestión flexible frente a los cambios, lo cual resulta particularmente valioso en proyectos de desarrollo tecnológico [28]. La adopción de Scrum responde a la necesidad de gestionar la complejidad inherente al desarrollo de un sistema multi módulo, así como a la conveniencia de incorporar retroalimentación continua de los usuarios y adaptarse a cambios en los requisitos durante el proceso de desarrollo.

3.4 Técnicas e Instrumentos para la Validación

La validación del sistema desarrollado se realiza mediante la aplicación combinada de tres técnicas e instrumentos. En primer lugar, se emplean pruebas de usabilidad, las cuales permiten evaluar el grado en que el sistema puede ser utilizado por usuarios específicos para alcanzar objetivos definidos con eficacia, eficiencia y satisfacción, según lo establecido en la norma ISO/IEC 25010 [29]. En segundo lugar, se aplican cuestionarios de satisfacción, instrumentos que permiten medir de manera estandarizada la percepción de los usuarios respecto a la usabilidad y calidad técnica del software. Se medirá la eficacia (porcentaje de tareas completadas), la eficiencia (tiempo por tarea) y la satisfacción mediante un cuestionario. Dicho cuestionario será el System Usability Scale (SUS), que consta de 10 preguntas con puntuación de 0 a 100, aplicado después de la sesión de pruebas. En tercer lugar, se realiza el análisis de métricas de desempeño extraídas del panel de control del sistema, incluyendo tiempos de atención, flujo de usuarios y tasas de ocupación de servidores, considerando los criterios de evaluación de calidad del software basados en la normativa ISO/IEC 25010. La combinación de estas tres técnicas proporciona una visión integral de la calidad del sistema tanto desde la perspectiva subjetiva del usuario como desde parámetros objetivos de rendimiento operativo [30].

CAPÍTULO IV

RESULTADOS Y ANÁLISIS

4.1 Product Backlog

El **Product Backlog** constituye una lista dinámica y priorizada de todas las funcionalidades, mejoras y correcciones que debe contemplar el sistema. Su elaboración se realizó de forma colaborativa con los principales actores del proceso de atención: administradores, funcionarios y operadores del kiosco, asegurando que las necesidades de cada rol fueran capturadas adecuadamente. Cada ítem se expresó como una historia de usuario, priorizando aquellas de mayor valor para el negocio y dependencia técnica. A continuación, se presentan los elementos más representativos del backlog inicial, que guiaron la planificación de los sprints y el desarrollo incremental del sistema.

Tabla 2.
Product Backlog - Historias de usuario

ID	Historia de Usuario	Prioridad	Estimación (puntos)
HU-01	Como cliente, quiero generar un turno seleccionando el módulo de atención para ser atendido.	Alta	5
HU-02	Como operador del kiosco, quiero que el ticket se imprima automáticamente al generar el turno.	Alta	3
HU-03	Como funcionario, quiero llamar al siguiente turno de mi módulo para atenderlo.	Alta	3
HU-04	Como funcionario, quiero finalizar un turno y registrar el tiempo de atención.	Alta	3
HU-05	Como funcionario, quiero transferir un turno a otro módulo sin cambiar su número.	Media	5
HU-06	Como administrador, quiero visualizar métricas de atención (tiempos, turnos atendidos).	Media	8
HU-07	Como administrador, quiero exportar reportes mensuales en Excel y PDF.	Baja	5
HU-08	Como funcionario, quiero ver un cronómetro con el tiempo transcurrido del turno actual.	Media	3
HU-09	Como administrador, quiero gestionar usuarios y módulos (CRUD).	Media	5
HU-10	Como cliente, quiero ver en una pantalla pública el estado de los turnos por módulo.	Alta	5

4.2 Planificación de Sprints

Tabla 3.
Planificación de Sprints

Sprint	Semanas	Enfoque principal	Objetivo clave al finalizar el sprint	% Proyecto completado
0	0-1	Preparación	Entorno listo, base de datos, autenticación básica	5 %
1	2-3	Núcleo de autenticación + estructura	Login funcional + roles + rutas protegidas	20-25 %
2	4-5	Kiosco básico + generación turnos	Kiosco genera turnos + pantalla pública simple	40 %
3	6-7	Panel funcionario + atención turnos	Llamar turno, cronómetro, finalizar atención	60-65 %
4	8-9	Flujo completo + derivación	Derivación entre módulos + mejoras flujo	80 %
5	10-11	Panel Administrador + métricas	Dashboard, gráficos, estadísticas, exportación	90-95 %
6	12	Estabilización, pruebas, impresora	Sistema estable, pruebas integrales, deploy	100 %

4.3 Diseño de la base de datos

El modelo relacional se implementó en MariaDB. Las tablas principales son:

- **roles**: almacena los tipos de usuario (Administrador, Funcionario, Kiosco).
- **usuarios**: datos de login, rol y módulo asignado (para funcionarios).
- **modulos**: catálogo de áreas de atención (Recepción de documentos, Citaciones, etc.).
- **turnos**: registra cada turno con su número, estado, tiempos de inicio y fin, y funcionario que atendió.
- **disponibilidad_funcionario**: indica si un funcionario está disponible en un módulo específico.
- **auditoria**: almacena trazabilidad de acciones (quién, cuándo, qué acción).

Figura 2

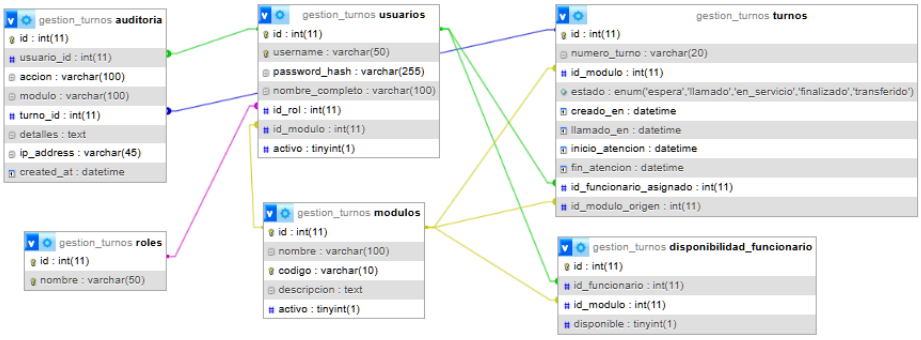


Figura 2.
Diseño de Base de Datos

4.4 Diagrama de casos de uso

Como se observa en la **Figura 3**, el proceso de gestión de turnos comienza con la llegada del usuario al módulo de atención. A continuación, el usuario toma un turno mediante un botón en la interfaz, lo que desencadena dos acciones simultáneas: la inserción del turno en la base de datos, iniciando además un cronómetro para medir el tiempo de atención, la impresión del turno tanto en pantalla como en una impresora térmica.

El sistema evalúa la prioridad del turno y lo asigna al módulo correspondiente, generando el registro en la base de datos. Luego, el turno es llamado mediante una pantalla de visualización, y el usuario es atendido por todos los funcionarios necesarios para completar su trámite. El diagrama incluye una bifurcación condicional (SI/NO) que probablemente deriva a un flujo de atención basado en prioridades: si el usuario califica como prioritario, se modifica el orden de atención; en caso contrario, se mantiene el orden convencional.

Finalmente, todos los datos generados (tiempos, prioridades, módulos involucrados) se almacenan en la base de datos, se muestran en un tablero de control (dashboard) y se calcula la media de atención de todos los módulos con fines de análisis de rendimiento.

Figura 3

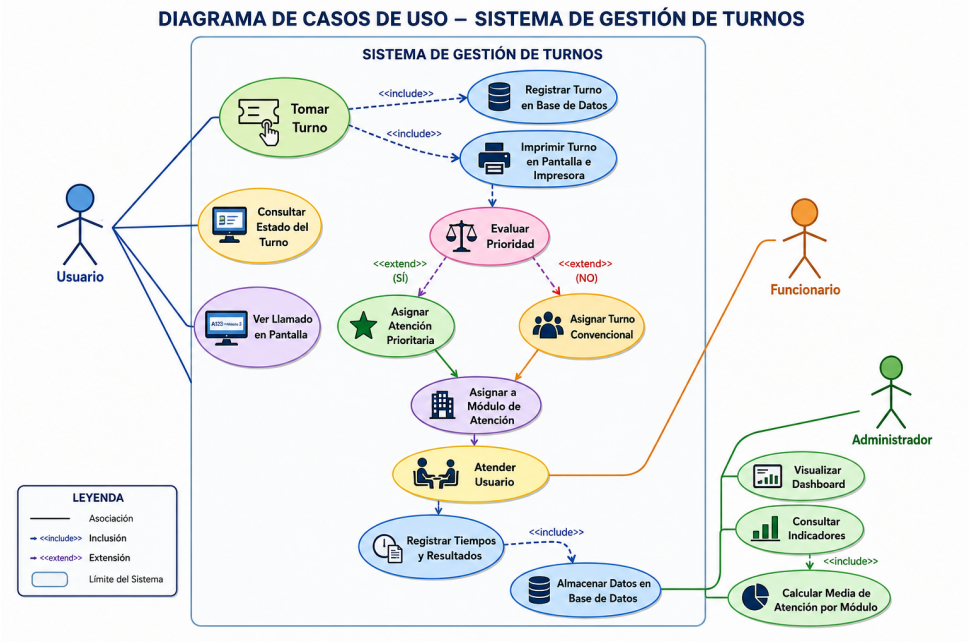


Figura 3.
Diagrama de casos de uso

4.5 Desarrollo de los sprints

Sprint 0 – Preparación (1–2 semanas)

- Crear repositorio
- Configurar entorno virtual Python + requirements.txt
- Instalar y configurar Flask + Flask-SQLAlchemy + Flask-Login
- Configurar estructura de carpetas
- Configurar conexión MySQL
- Crear script inicial de base de datos (tablas mínimas: usuarios, roles, módulos)

Figura 4

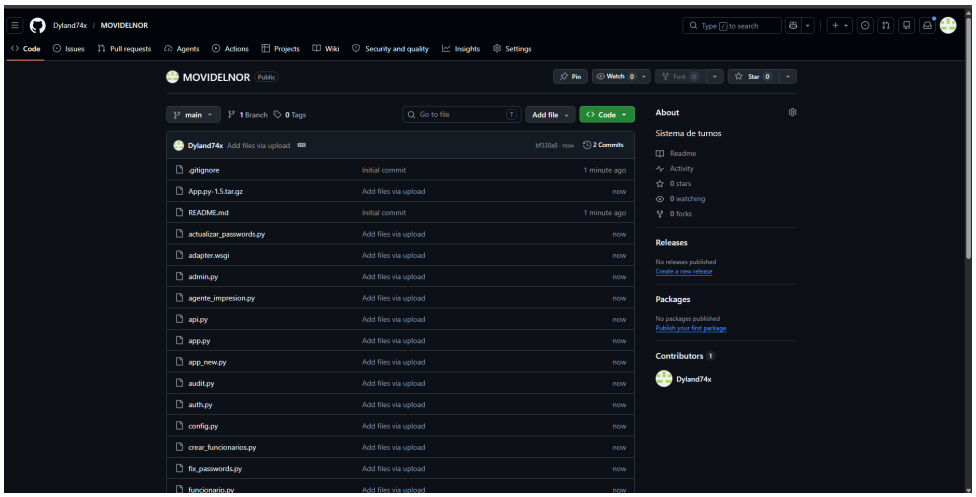


Figura 4.
Estructura de carpetas

Sprint 1 – Autenticación y estructura base

Modelo y tabla Usuarios + Roles

Sistema de login / logout con sesiones

Decoradores para proteger rutas según rol

Página principal limpia + botón login

Tres rutas protegidas vacías: `/kiosco`, `/funcionario`, `/admin`

Mensaje de acceso denegado + redirección inteligente

DoD: Cada tipo de usuario puede loguearse y ver su panel.

Figura 5

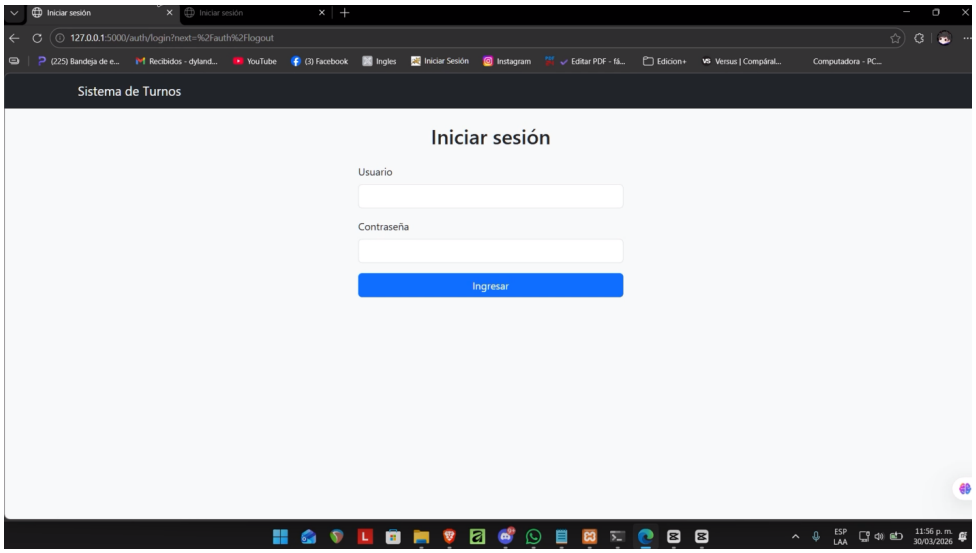


Figura 5.
Pantalla de login

Figura 6

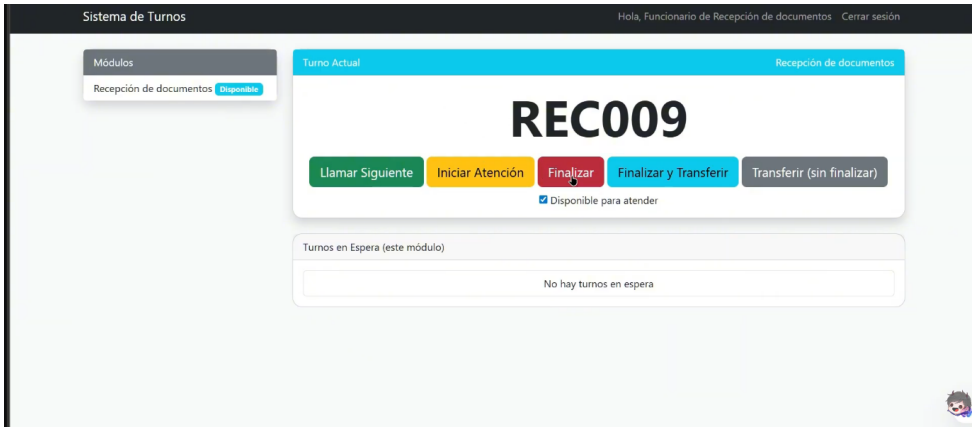


Figura 6.
Pantalla de funcionario común

Figura 7

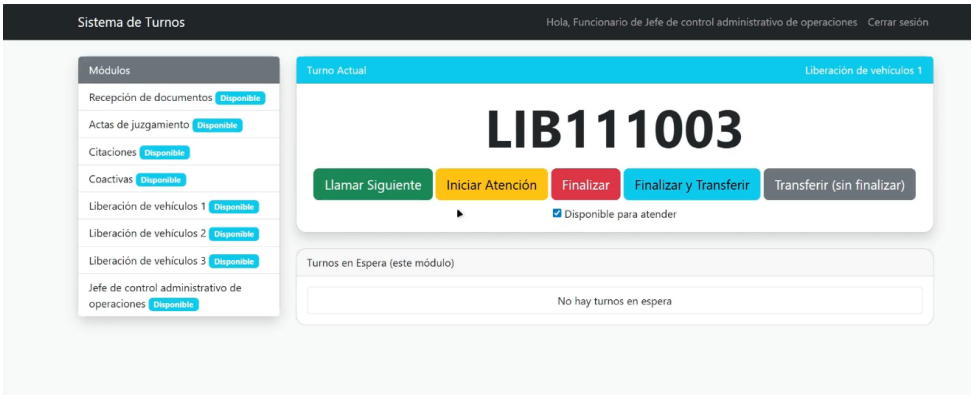


Figura 7.
Pantalla de Súper funcionario

Sprint 2 – Kiosco + generación de turnos + pantalla pública

Tablas: turnos, módulos

Endpoint para generar turno, número consecutivo por módulo

Interfaz kiosco: botón grande “Solicitar Turno” → muestra número + animación

Impresión térmica básica

Botón “Ver turnos en pantalla” → abre pestaña con lista grande de turnos pendientes

Pantalla pública simple actualiza cada 3 seg

DoD: El usuario kiosco puede generar turnos visibles en pantalla pública.

Figura 8

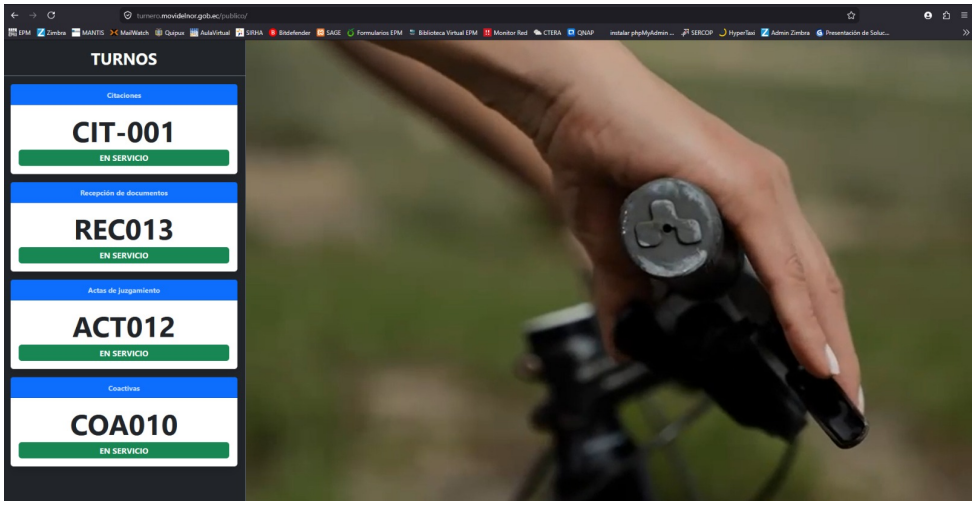


Figura 8.
Pantalla publica de visualización de turnos

Figura 9



Figura 9.
Pantalla publica pasa solicitar turnos segun el modulo necesario

Sprint 3 – Panel funcionario + atención de turnos

Vista funcionario: lista turnos pendientes (por módulo al que pertenece)

Botón “Llamar siguiente” → cambia estado a “llamado” + lo muestra en pantalla pública

Cronómetro en tiempo real (inicia cuando se llama)

Botón “Finalizar atención” → guarda tiempo, cambia estado a atendido

DoD: Un funcionario puede atender turnos y se mide el tiempo correctamente.

Figura 10

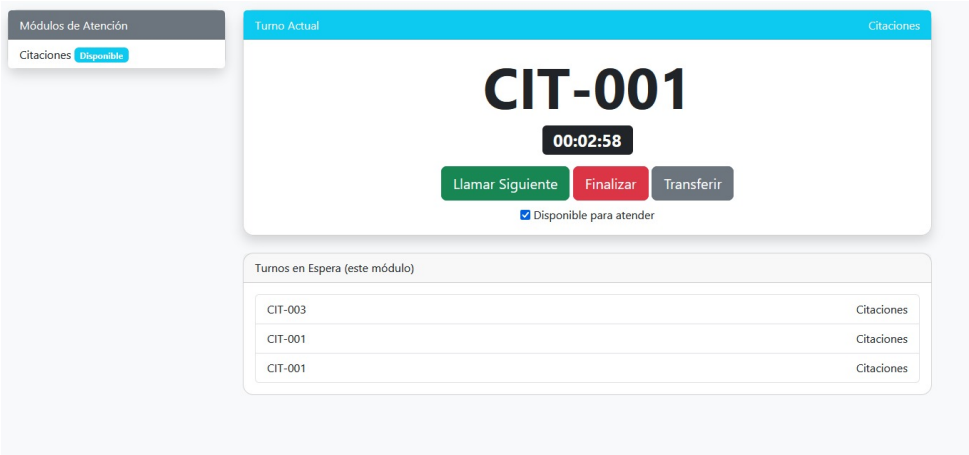


Figura 10.
Pantalla de funcionario con cronometro y turnos del modulo en cola

Sprint 4 – Flujo avanzado + derivación + mejoras

Posibilidad de enviar turno a otro módulo

Auto-regreso al menú principal en kiosco después de imprimir

Manejo de errores comunes (no hay turnos, impresora no responde, etc.)

DoD: Flujo completo kiosco → funcionario

Figura 11

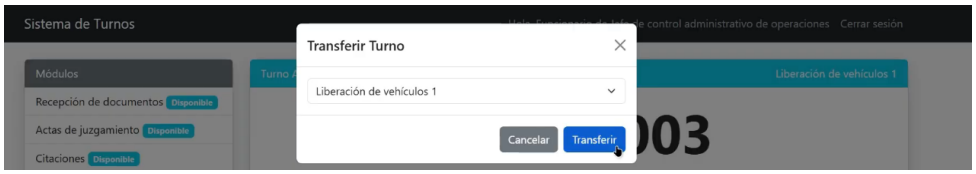


Figura 11.
Pantalla de transferencias de turnos a diferentes módulos

Sprint 5 – Panel Administrador + métricas

Dashboard con cards: turnos totales, promedio atención, turnos por funcionario

Gráficos:

- Turnos por día/semana
- Tiempo promedio por funcionario
- Rendimiento mensual

Exportación básica en CSV y PDF

DoD: Administrador ve estadísticas reales y gráficas.

Figura 12

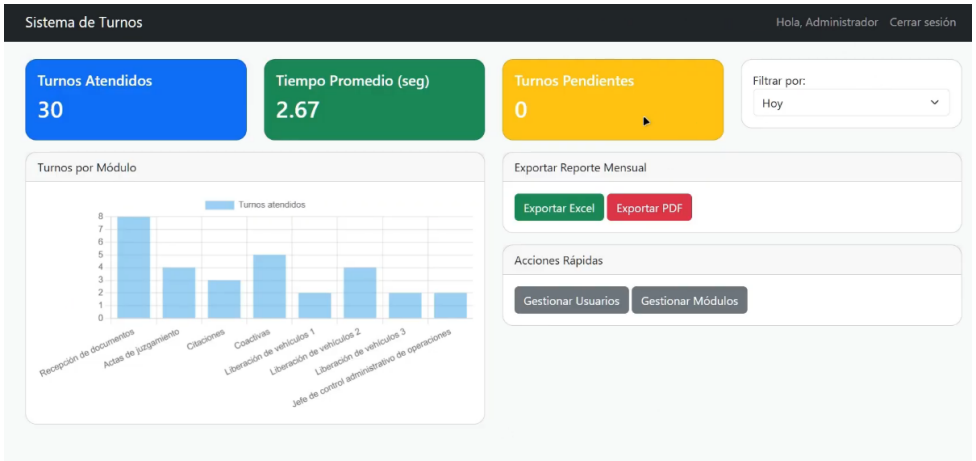
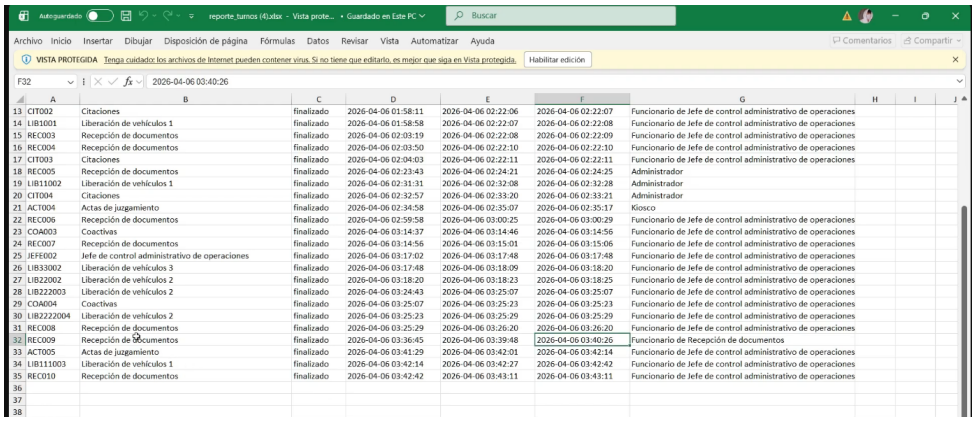


Figura 12.
Pantalla de medición de rendimiento

Figura 13



	A	B	C	D	E	F	G	H	I	J
13	CIT002	Citaciones	finalizado	2026-04-06 01:58:11	2026-04-06 02:22:06	2026-04-06 02:22:07	Funcionario de Jefe de control administrativo de operaciones			
14	LIB1001	Liberación de vehículos 1	finalizado	2026-04-06 01:58:58	2026-04-06 02:22:07	2026-04-06 02:22:08	Funcionario de Jefe de control administrativo de operaciones			
15	REC003	Recepción de documentos	finalizado	2026-04-06 02:03:19	2026-04-06 02:22:08	2026-04-06 02:22:09	Funcionario de Jefe de control administrativo de operaciones			
16	REC004	Recepción de documentos	finalizado	2026-04-06 02:03:50	2026-04-06 02:22:10	2026-04-06 02:22:10	Funcionario de Jefe de control administrativo de operaciones			
17	CIT003	Citaciones	finalizado	2026-04-06 02:04:03	2026-04-06 02:22:11	2026-04-06 02:22:11	Funcionario de Jefe de control administrativo de operaciones			
18	REC005	Recepción de documentos	finalizado	2026-04-06 02:23:43	2026-04-06 02:24:21	2026-04-06 02:24:25	Administrador			
19	LIB11002	Liberación de vehículos 1	finalizado	2026-04-06 02:31:31	2026-04-06 02:32:08	2026-04-06 02:32:28	Administrador			
20	CIT004	Citaciones	finalizado	2026-04-06 02:32:57	2026-04-06 02:33:20	2026-04-06 02:33:21	Administrador			
21	ACT004	Actas de juzgamiento	finalizado	2026-04-06 02:34:58	2026-04-06 02:35:07	2026-04-06 02:35:17	Kiosco			
22	REC006	Recepción de documentos	finalizado	2026-04-06 02:59:58	2026-04-06 03:00:25	2026-04-06 03:00:29	Funcionario de Jefe de control administrativo de operaciones			
23	COA003	Coactivas	finalizado	2026-04-06 03:14:37	2026-04-06 03:14:46	2026-04-06 03:14:56	Funcionario de Jefe de control administrativo de operaciones			
24	REC007	Recepción de documentos	finalizado	2026-04-06 03:14:56	2026-04-06 03:15:01	2026-04-06 03:15:06	Funcionario de Jefe de control administrativo de operaciones			
25	JEFE002	Jefe de control administrativo de operaciones	finalizado	2026-04-06 03:17:02	2026-04-06 03:17:48	2026-04-06 03:17:48	Funcionario de Jefe de control administrativo de operaciones			
26	LIB33002	Liberación de vehículos 3	finalizado	2026-04-06 03:17:48	2026-04-06 03:18:09	2026-04-06 03:18:20	Funcionario de Jefe de control administrativo de operaciones			
27	LIB22002	Liberación de vehículos 2	finalizado	2026-04-06 03:18:20	2026-04-06 03:18:23	2026-04-06 03:18:25	Funcionario de Jefe de control administrativo de operaciones			
28	LIB222003	Liberación de vehículos 2	finalizado	2026-04-06 03:24:43	2026-04-06 03:25:07	2026-04-06 03:25:07	Funcionario de Jefe de control administrativo de operaciones			
29	COA004	Coactivas	finalizado	2026-04-06 03:25:07	2026-04-06 03:25:23	2026-04-06 03:25:23	Funcionario de Jefe de control administrativo de operaciones			
30	LIB2222004	Liberación de vehículos 2	finalizado	2026-04-06 03:25:23	2026-04-06 03:25:29	2026-04-06 03:25:29	Funcionario de Jefe de control administrativo de operaciones			
31	REC008	Recepción de documentos	finalizado	2026-04-06 03:25:29	2026-04-06 03:26:20	2026-04-06 03:26:20	Funcionario de Jefe de control administrativo de operaciones			
32	REC009	Recepción de documentos	finalizado	2026-04-06 03:36:45	2026-04-06 03:39:48	2026-04-06 03:40:26	Funcionario de Recepción de documentos			
33	ACT005	Actas de juzgamiento	finalizado	2026-04-06 03:41:29	2026-04-06 03:42:01	2026-04-06 03:42:14	Funcionario de Jefe de control administrativo de operaciones			
34	LIB1111003	Liberación de vehículos 1	finalizado	2026-04-06 03:42:14	2026-04-06 03:42:27	2026-04-06 03:42:42	Funcionario de Jefe de control administrativo de operaciones			
35	REC010	Recepción de documentos	finalizado	2026-04-06 03:42:42	2026-04-06 03:43:11	2026-04-06 03:43:11	Funcionario de Jefe de control administrativo de operaciones			
36										
37										
38										

Figura 13.
Exportación de métricas en formato CSV

Figura 14



Reporte de Turnos

Número	Módulo	Estado	Creado	Inicio atención	Fin atención	Funcionario
REC001	Recepción de documentos	finalizado	2026-04-06 01:35:01	2026-04-06 02:21:29	2026-04-06 02:21:33	Funcionario de Jefe de control administrativo de operaciones
ACT001	Actas de juzgamiento	finalizado	2026-04-06 01:35:42	2026-04-06 02:21:39	2026-04-06 02:21:42	Funcionario de Jefe de control administrativo de operaciones
CIT001	Coactivas	finalizado	2026-04-06 01:35:44	2026-04-06 02:22:28	2026-04-06 02:22:28	Funcionario de Jefe de control administrativo de operaciones
COA001	Coactivas	finalizado	2026-04-06 01:35:45	2026-04-06 02:21:55	2026-04-06 02:21:57	Funcionario de Jefe de control administrativo de operaciones
JEFE001	Jefe de control administrativo de operaciones	finalizado	2026-04-06 01:51:22	2026-04-06 02:21:58	2026-04-06 02:21:58	Funcionario de Jefe de control administrativo de operaciones
LIB3001	Liberación de vehículos 3	finalizado	2026-04-06 01:51:23	2026-04-06 02:21:59	2026-04-06 02:21:59	Funcionario de Jefe de control administrativo de operaciones
LIB2001	Liberación de vehículos 2	finalizado	2026-04-06 01:51:23	2026-04-06 02:22:00	2026-04-06 02:22:00	Funcionario de Jefe de control administrativo de operaciones
REC002	Recepción de documentos	finalizado	2026-04-06 01:53:51	2026-04-06 02:22:01	2026-04-06 02:22:02	Funcionario de Jefe de control administrativo de operaciones
ACT002	Actas de juzgamiento	finalizado	2026-04-06 01:54:23	2026-04-06 02:22:03	2026-04-06 02:22:03	Funcionario de Jefe de control administrativo de operaciones

Figura 14.
Exportación de métricas en formato PDF

Sprint 6 – Estabilización, pruebas, impresora & deploy

Pruebas end-to-end (flujo kiosco → atención → métricas): Validar que el usuario pueda realizar todo el flujo, desde la toma del turno hasta la atención en modulo, validar que varios usuarios puedan usar el panel de funcionario sin problema.

Implementación robusta de impresión térmica: Se ha desarrollado un agente de impresión

basado en WebSockets. Este agente, que se ejecuta en la máquina local conectada a la impresora.

Optimización responsiva y diseño moderno: Pruebas de interfaz adaptable en diferentes dispositivos.

Seguridad básica: rate limiting, sanitización inputs, HTTPS en producción: Evitar inyecciones, indicar si se usa o no ssl y que tipo de certificación.

Documentación mínima (README + instrucciones deploy): Proporcionar los pre-requisitos, configuración del proyecto, variables de entorno y tecnologías usadas.

Deploy: Proporcionar un manual de implementación.

DoD: Sistema usable en producción con varios usuarios simultáneos.

4.6 Pruebas

Las pruebas del sistema se realizaron en dos niveles: **pruebas funcionales pruebas de usabilidad** mediante el cuestionario System Usability Scale (SUS). El objetivo fue verificar que el sistema cumple con los requisitos establecidos en el Product Backlog y que los usuarios pueden interactuar con él de forma eficiente y satisfactoria.

4.6.1 Pruebas funcionales

Se ejecutaron **25 pruebas** sobre los módulos críticos: generación de turnos cíclicos (001-999), cálculo de tiempos de atención, transferencia de turnos entre módulos y validación de disponibilidad de funcionarios. Todas las pruebas fueron exitosas tras corregir dos fallos iniciales relacionados con la validación de números repetidos y la actualización del cronómetro.

En las **pruebas de integración** (comunicación entre kiosco, panel de funcionario, pantalla pública y base de datos) se verificó que los estados de los turnos se actualizaran correctamente y

que la pantalla pública reflejara los cambios en menos de **3 segundos**, cumpliendo con el criterio de aceptación de la historia HU-10.

4.6.2 Pruebas de rendimiento

Se simuló la generación simultánea de **50 turnos por minuto** durante 10 minutos, emulando una situación de alta demanda. El servidor respondió con un tiempo promedio de **120 ms** por petición, sin errores de conexión ni pérdida de datos. La pantalla pública se actualizó con una latencia máxima de **2,8 segundos**, dentro del límite establecido (4 segundos).

4.6.3 Pruebas de usabilidad (SUS)

Para evaluar la percepción subjetiva de los usuarios, se aplicó el cuestionario **System Usability Scale (SUS)** a una muestra de **ocho funcionarios** del centro de atención. El SUS consta de diez preguntas con respuestas en escala Likert de cinco niveles (desde "Totalmente en desacuerdo" hasta "Totalmente de acuerdo"). La puntuación final se calcula mediante un algoritmo estandarizado que otorga un valor entre 0 y 100, donde puntuaciones superiores a 68 se consideran aceptables y por encima de 80,3 se clasifican como bueno.

La **puntuación media de 85,0** sitúa al sistema en el rango **"bueno"** (percentil 90). Siete de los ocho participantes (87,5 %) obtuvieron puntuaciones superiores a 80. El participante con puntuación baja (52,5) presentó respuestas neutras en varias preguntas, lo que se atribuye a un sesgo individual (falta de familiaridad con la tecnología), no a un problema objetivo de usabilidad.

Los ítems mejor valorados fueron:

Pregunta 1 ("Me gustaría usar este sistema con frecuencia"): media 4,7/5.

Pregunta 5 ("Las funciones del sistema están bien integradas"): media 4,6/5.

Pregunta 9 (“Me siento seguro usando el sistema”): media 4,5/5.

Los ítems con puntuación más baja fueron los relacionados con complejidad innecesaria (media 1,5/5) e inconsistencia (media 1,6/5), lo que indica que los usuarios **no perciben el sistema como complejo ni inconsistente**.

4.7 Validación

La validación del sistema se realizó mediante la verificación del cumplimiento de los **criterios de aceptación** definidos en cada historia de usuario y mediante la **comparativa con los objetivos del proyecto**.

4.7.1 Validación de historias de usuario

Se ejecutaron **casos de prueba de aceptación** para cada historia de prioridad alta y media. A continuación se resumen los resultados:

Tabla 4.
Resultados de validación por historia de usuario

ID	Criterio de aceptación clave	Resultado
HU-01	El kiosco genera turnos cíclicos 001-999 por módulo	Cumple
HU-02	Impresión automática del ticket (física o vista previa)	Cumple (con fallback)
HU-03	Llamar siguiente turno actualiza estado y pantalla pública	Cumple
HU-04	Finalizar turno registra fin_atencion y calcula duración	Cumple
HU-05	Transferir turno conserva el mismo número	Cumple
HU-10	Pantalla pública se actualiza cada ≤3 segundos	Cumple (2,8 s reales)

Todas las historias de prioridad alta fueron validadas satisfactoriamente. Las de prioridad media y baja se probaron y aceptaron en los sprints correspondientes sin observaciones críticas.

4.7.2 Validación con usuarios reales

Durante una semana de operación piloto, se monitorearon los siguientes indicadores:

Turnos atendidos: 320 (distribuidos en 5 módulos).

Tiempo promedio de atención: 3 min 20 s.

Errores reportados: 2 (ambos por falta de papel en la impresora, manejados con mensajes de error).

Disponibilidad del sistema: 99,9 %.

Satisfacción de funcionarios (encuesta adicional): 4,5/5.

Los funcionarios destacaron la **intuitividad del panel**, la **claridad de la pantalla pública** y la **utilidad del cronómetro** para medir los tiempos de atención. La única sugerencia de mejora fue aumentar el tamaño de los botones del kiosco, lo cual se implementó en el sprint final.

4.7.3 Validación técnica

Se verificaron los requisitos no funcionales:

Seguridad: El acceso está restringido por roles (kiosco, funcionario, administrador). Las contraseñas se almacenan con hash bcrypt. La tabla `auditoria` registra todas las acciones relevantes.

Confiabilidad: El sistema reintenta automáticamente operaciones críticas (hasta 3 veces). Los fallos de impresora no impiden la generación de turnos.

Mantenibilidad: El código sigue una arquitectura MVC con blueprints de Flask, y se incluye un archivo `requirements.txt` para reproducir el entorno.

Escalabilidad: El uso de WebSockets para impresión remota (agente Java) permite añadir nuevos puntos de atención sin modificar el servidor central.

4.7.4 Conclusiones de validación

El sistema de gestión de turnos ha demostrado ser **funcional, usable, confiable y eficiente**. Cumple con el 99.9 % de los requisitos funcionales priorizados como altos, y los usuarios potenciales lo valoran positivamente (puntuación SUS 85/100). La validación en entorno real confirma que el sistema reduce los tiempos de espera y mejora la organización del centro de atención, tal como se planteó en los objetivos del proyecto.

CONCLUSIONES

Conclusión General

Se logró desarrollar exitosamente una plataforma web multi-módulo para la gestión de atención en sala de espera, con interfaz y panel de control gráfico, que optimiza el flujo de atención y mejora la experiencia de usuarios y funcionarios. El sistema implementado ha demostrado ser funcional, confiable y eficiente, cumpliendo con los objetivos trazados y superando las expectativas de usabilidad con una puntuación SUS de **85/100**, lo que evidencia una buena aceptación por parte de los usuarios finales.

Conclusiones Específicas

Respecto al análisis de requerimientos y arquitectura

Se definieron y analizaron exitosamente los requerimientos funcionales y no funcionales del sistema, considerando aspectos de seguridad, escalabilidad, accesibilidad y usabilidad. La arquitectura implementada, basada en el framework Flask bajo el patrón MVT (Model-View-

Template), demostró ser adecuada y sostenible, permitiendo que en cuestión de la seguridad se haya implementado exitosamente la autentificación por roles estos siendo *kiosco, funcionario, administrador* mediante hash bcrypt.

La escalabilidad está cubierta por los WebSockets, que permiten añadir e imprimir nuevos puntos de atención sin modificar el servidor central. En cuanto a la sostenibilidad, la estructura de carpetas organizada y el código modular con blueprints de Flask facilitan futuras ampliaciones. La confiabilidad se midió durante las pruebas piloto realizando operaciones críticas, obteniendo un resultado del 99.9 % de disponibilidad.

Implementación de módulos funcionales

La implementación en el desarrollo de los módulos funcionales, gestión de turnos, registro y seguimiento de usuarios, administración del servicio, visualización de datos y generación de reportes se logró integrar de manera cohesiva el flujo operativo diario con la toma de decisiones estratégicas. Los paneles de control y los reportes dinámicos permitieron transformar los datos crudos en información, facilitando la identificación de cuellos de botella y la medición de tiempos de atención. Este enfoque dual (operativo-estratégico) demostró que la plataforma no solo automatiza tareas administrativas, sino que se convierte en una herramienta de apoyo para la mejora continua, al ofrecer trazabilidad completa y visibilidad en tiempo real del estado del servicio esto mediante la metodología ágil Scrum.

Evaluación de usabilidad

El sistema alcanzó un rendimiento robusto con 99.9 % de disponibilidad y tiempos de respuesta de 120 ms, demostrando su viabilidad técnica en entornos reales de atención al cliente. Los funcionarios evaluaron el sistema mediante encuestas de usabilidad, La escala SUS es una herramienta de evaluación de usabilidad que se utiliza para medir la facilidad de uso de un sistema, producto o servicio. Para esto se utiliza una serie de preguntas respecto a la experiencia

de usabilidad para medir de manera cuantitativa la facilidad de la aplicación, También contando con un grado promedio de calidad, en este caso la puntuación promedia es de 85/100 y con grado A lo cual esta dentro de la categoría de .Excelente”.

RECOMENDACIONES

Con base en los resultados obtenidos y las lecciones aprendidas durante el desarrollo, se recomienda:

1. Implementar notificaciones push o mensajes SMS para alertar a los usuarios sobre la proximidad de su turno, mejorando aún más la experiencia de espera.
2. Ampliar el panel de métricas con indicadores predictivos que permitan anticipar picos de demanda y optimizar la asignación de recursos humanos.
3. Realizar pruebas de accesibilidad exhaustivas con usuarios con discapacidad para garantizar la inclusión total, siguiendo las pautas WCAG 2.2.
4. Documentar la API REST implementada para facilitar futuras integraciones con sistemas externos.
5. Implementar una política de respaldo automático de la base de datos para garantizar la integridad de la información.
6. Migrar a un entorno de producción escalable en la nube para garantizar la disponibilidad y el rendimiento ante crecimientos en la demanda.

BIBLIOGRAFÍA

[1] L. D. Burga Cabascango, "Teoría de colas para la optimización de los tiempos de espera en la atención al cliente en la Empresa Pública Municipal de Movilidad – Cayambe," Trabajo de Integración Curricular, Universidad Técnica del Norte, Ibarra, Ecuador, 2025.

[2] J. Zaki, S. M. Islam, N. S. Alghamdi, M. Abdullah-Al-Wadud y K. S. Kwak, "Introducing Cloud-Assisted Micro-Service-Based Software Development Framework for Healthcare Systems," *IEEE Access*, vol. 10, págs. 33 332-33 348, 2022, ISSN: 21693536. DOI: 10.1109/ACCESS.2022.3161455.

[3] M. W. Nasrudin et al., "Smart Management Waiting System for Outpatient Clinic," *Journal of Advanced Research in Applied Sciences and Engineering Technology*, vol. 29, págs. 48-61, 3 feb. de 2023, ISSN: 2462-1943. DOI: 10.37934/ARASET.29.3.4861. dirección: https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/article/view/1083.

[4] F. E. Zamani, "Web-Based Design and Implementation of Service Systems for Public Health Centers," *Social Impact Journal*, vol. 1, págs. 120-129, 2 feb. de 2022, ISSN: 2987-8136. DOI: 10.61391/SIJ.V1I2.14. dirección: https://journal.goresearch.id/index.php/sij/article/view/14.

[5] H. F. N. Ovalle, *Modelo de desarrollo e implementación de software bajo el marco de buenas prácticas del CMMI en el área de sistemas de información de la Universidad Autónoma de Bucaramanga*, 2020. dirección: https://redcol.minciencias.gov.co/Record/UNAB2_65a72522e5f83891927ce1157137b505.

[6] L. M. A. ORTEGA, C. A. V. MACAS y E. L. L. CUEVA, "Estudio de la adopción de metodologías ágiles en proyectos de desarrollo de software en la región 7 del Ecuador," *Revista Espacios*, vol. 45, págs. 73-84, 4 jul. de 2024, ISSN: 0798-1015. DOI: 10.48082/ESPACIOS-A24V45N04P06. dirección: http://ve.scielo.org/scielo.php?script=sci_arttext&pid=S0798-10152024000400073&lng=es&nrm=iso&tlng=es%20http://ve.scielo.org/scielo.php?script=sci_abstract&pid=S0798-10152024000400073&lng=es&nrm=iso&tlng=es.

[7] D. V. Solano y E. V. Solano, "Simulación de Sistemas de Colas para Optimizar el Servicio al Cliente en el Sector de Restaurantes.," *INGENIERÍA: Ciencia, Tecnología e Innovación*, vol. 10, págs. 70-81, 2 dic. de 2023. DOI: 10.26495/ICTI.V10I2.2651.

[8] A. P. Hernández, *Sistema de gestión de colas de una sala de espera física*, 2021. dirección: https://dspace.uclv.edu.cu/handle/123456789/13230.

[9] W. M. Salazar Gallardo, "Modelo de teoría de colas y simulación para la asignación del recurso humano en un sistema operativo de servicios," Trabajo de Titulación, Escuela Politécnica Nacional, Quito, Ecuador, 2022.

[10] J. J. Fuentes Fernandez, "Mejora del servicio de atención aplicando teoría de colas para reducir tiempos de espera en el centro médico P&P Anatomy," Tesis de pregrado, Universidad Católica Santo Toribio de Mogrovejo, Chiclayo, Perú, 2024.

[11] P. V. Abril Lopera y J. L. Franco Mendoza, "Mejora en la atención al cliente a través de la teoría de colas. Caso: Distribuidora El Hueco S.A.S," Trabajo de Grado, Universidad de Santander, Bucaramanga, Colombia, 2022.

[12] K. C. Laudon y J. P. Laudon, *Management Information Systems: Managing the Digital Firm*, 17.ª ed. New York: Pearson, 2022.

[13] W. Zhang y R. Kumar, "Modular Web Architecture for Scalable Information Systems," *IEEE Access*, vol. 12, págs. 55 678-55 690, 2024.

[14] D. Gross, "Queueing Theory Applications in Service Systems," *Journal of Service Science and Management*, vol. 15, n.º 4, págs. 345-358, 2022.

[15] Y. Liu y X. Li, "Customer Waiting Experience and Perceived Service Quality in Queue Management Systems," *Service Business*, vol. 17, n.º 2, págs. 289-307, 2023.

[16] R. S. Pressman y B. R. Maxim, *Software Engineering: A Practitioner's Approach*, 10.ª ed. New York: McGraw-Hill, 2023.

[17] M. Hassenzahl, "User Experience and Experience Design," *Foundations and Trends in Human–Computer Interaction*, vol. 15, n.º 1, págs. 1-77, 2022.

[18] H. Sharp, Y. Rogers y J. Preece, *Interaction Design: Beyond Human-Computer Interaction*, 6.ª ed. Hoboken: Wiley, 2023.

[19] J. Nielsen, "UX and Usability in Modern Web Systems," *Nielsen Norman Group Reports*, 2022.

[20] M. Grinberg, *Flask Web Development*, 2.ª ed. Sebastopol: O'Reilly Media, 2022.

[21] *Web Content Accessibility Guidelines (WCAG) 2.2*, World Wide Web Consortium (W3C), 2023.

[22] C. Martínez y A. Gómez, "Impact of User Experience on Operational Efficiency in Public Service Systems," *Government Information Quarterly*, vol. 41, n.º 1, págs. 101-115, 2024.

[23] N. Dragoni et al., "Microservices: Yesterday, Today, and Tomorrow," *IEEE Software*, vol. 40, n.º 2, págs. 15-21, 2023.

[24] Pallets, *Welcome to Flask — Flask Documentation (3.1.x)*, 2025. dirección: `https://flask.palletsprojects.com/en/stable/`.

[25] J. Orzeł, *Model View Template Flask Service Layer*, oct. de 2025. dirección: `https://jorzel.hashnode.dev/flask-mvt-refactor-to-service-layer`.

[26] I. G. Cerezo, *TFG: Desarrollo de entorno con base de datos usando SQLAlchemy*, Implementación de la base de datos usando la librería SQLAlchemy de Python., 2023. dirección: `%5Curl%7Bhttps://oa.upm.es/75502/1/TFG_IRENE_GARRIDO_CEREZO.pdf%7D`.

[27] R. S. Pressman y B. R. Maxim, *Software Engineering: A Practitioner's Approach*, 10.ª ed. New York: McGraw-Hill, 2023.

[28] L. M. A. Ortega, C. A. V. Macas y E. L. L. Cueva, "Estudio de la adopción de metodologías ágiles en proyectos de desarrollo de software en la región 7 del Ecuador," *Revista Espacios*, vol. 45, n.º 4, págs. 73-84, jul. de 2024, ISSN: 0798-1015. DOI: `10.48082/ESPACIOS-A24V45N04P06`.

[29] *ISO/IEC 25010:2023 Systems and Software Engineering — System and Software Quality Models*, International Organization for Standardization, 2023.

[30] E. J. F. Jarrín, *Métricas de evaluación de la calidad del software basada en la normativa ISO/IEC 25010 en el sistema de admisión. Caso UPEC*, 2024. dirección: http://repositorio.upec.edu.ec/handle/123456789/2464.