UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

POSGRADO

MAESTRÍA EN INGENIERÍA DE SOFTWARE

Marco tecnológico con el uso de stack MEAN aplicado a la gestión de turnos en la Cooperativa de Ahorro y Crédito Tulcán Ltda.

Trabajo de titulación previa la obtención del
Título de Magister en Ingeniería de Software

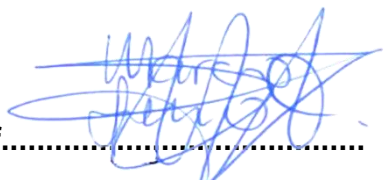
Autor: Iván Darío Rojas Rojas
Tutor: MSc. Revelo Aldás Marco David

Tulcán, 2025

CERTIFICADO DEL TUTOR

Certifico que el maestrante Rojas Rojas Iván Darío con el número de cédula 0401709035 ha elaborado el trabajo de titulación: “Marco tecnológico con el uso de stack MEAN aplicado a la gestión de turnos en la Cooperativa de Ahorro y Crédito Tulcán Ltda.”.

Este trabajo se sujeta a las normas y metodologías dispuestas en el Reglamento de la Unidad de Titulación de Postgrado con RESOLUCIÓN Nº 150-CSUP- 2020, por lo tanto, autorizo su presentación para la sustentación respectiva



f.....

MSc. Revelo Aldás Marco David

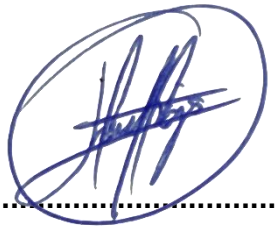
TUTOR

Tulcán, abril de 2025

AUTORÍA DE TRABAJO

El presente trabajo de titulación constituye un requisito previo para la obtención del título de Magister en Ingeniería de Software.

Yo, Rojas Rojas Iván Darío con cédula de identidad número 0401709035 declaro: que la investigación es absolutamente original, auténtica, personal y los resultados y conclusiones a los que he llegado son de mi absoluta responsabilidad.



f.....

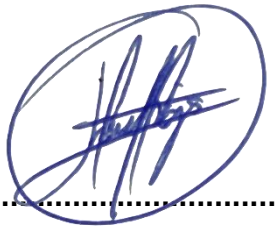
Rojas Rojas Iván Darío

AUTOR

Tulcán, abril de 2025

ACTA DE CESIÓN DE DERECHOS DEL TRABAJO DE TITULACIÓN

Yo, Rojas Rojas Iván Darío declaro ser autor de los criterios emitidos en el trabajo de titulación: “Marco tecnológico con el uso de stack MEAN aplicado a la gestión de turnos en la Cooperativa de Ahorro y Crédito Tulcán Ltda.” y eximo expresamente a la Universidad Politécnica Estatal del Carchi y a sus representantes legales de posibles reclamos o acciones legales.



f.....

Rojas Rojas Iván Darío

AUTOR

Tulcán, abril de 2025

AGRADECIMIENTO

Durante el transcurso del desarrollo de este trabajo de grado, han existido personas que me han apoyado de gran manera y en diferentes aspectos con su conocimiento, experticia y brindado su valiosa ayuda, pero también han contribuido de manera igualmente significativa con su apoyo emocional y motivacional, agradezco sinceramente a todas esas personas.

Primero, quiero agradecer profundamente a mi familia, mi madre por su guía, compañía y confianza eterna con la que ha logrado proveerme de valores que han sido fundamentales para superar todos los desafíos académicos de este proyecto y de mi vida en general, a mi hermana que siempre ha sido un apoyo fundamental e incondicional durante gran parte de mi vida y carrera académica y profesional, y a mis niños que como siempre me han llenado de valentía y coraje.

A mi tutor por su guía y valiosas aportaciones que permitieron que este proyecto se realizara de la mejor manera posible, lo cual me ha enriquecido a nivel intelectual y personal.

Finalmente agradezco a la Universidad Politécnica Estatal del Carchi y a la Cooperativa de Ahorro y Crédito Tulcán Ltda., por proveerme del entorno y de todos los recursos que fueron necesarios para la realización exitosa de este trabajo de grado.

Con gratitud,

Iván Darío Rojas Rojas

DEDICATORIA

Dedico este trabajo de titulación a mi familia, amigos y a mis niños, que con paciencia y cariño me han sabido acompañar en cada alto y bajo que se ha presentado durante el desarrollo de este trabajo. Gracias por su incondicional apoyo, por su paciencia y la motivación constante que me brindan; cada paso y triunfo que logre será por y para ustedes, pues sin ustedes no lo lograría.

Iván Darío Rojas Rojas

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INDICE DE ANEXOS

RESUMEN

Esta investigación tuvo como objetivo proponer un marco tecnológico con el uso del stack MEAN aplicado a la gestión de turnos en la Cooperativa de Ahorro y Crédito Tulcán Ltda. El estudio tuvo un enfoque mixto de tipo descriptiva y de campo. Se describieron los procesos actuales de gestión de turnos en diez cooperativas de ahorro y crédito, abarcando un análisis de la Cooperativa de Ahorro y Crédito Tulcán Ltda. Además, se evalúa el impacto operativo de estos sistemas en indicadores como tiempos de espera y eficiencia en la atención. La información recolectada fue procesada mediante herramientas como Excel, aplicando técnicas estadísticas descriptivas y correlacionales. Se desarrollo un diseño de un sistema con el stack MEAN. Finalmente se formuló un marco tecnológico integral, para la gestión de turnos, fundamentado en el stack MEAN: MongoDB, Express.js, Angular y Node.js, complementado con Socket.io para comunicación en tiempo real. El estudio identificó que los métodos tradicionales de gestión generaban ineficiencia operativa, tiempos de espera prolongados e insatisfacción del cliente. Mediante una metodología y el uso de Scrum para el desarrollo, se implementó un sistema web que incluye toma de turnos presenciales y en línea con validación OTP, transferencia entre módulos, notificaciones en tiempo real mediante Socket.io, generación de reportes y tickets con código QR. La arquitectura propuesta demostró ventajas en escalabilidad y mantenibilidad al utilizar JavaScript en toda la stack tecnológica. La implementación fue validada con usuarios finales, resolvió las deficiencias identificadas y estableció bases para futuros desarrollos tecnológicos en la cooperativa. El marco tecnológico con stack MEAN se configura como una solución efectiva para la modernización de procesos en instituciones financieras.

Palabras clave: arquitectura de software, cooperativas, digitalización, gestión de turnos, Stack MEAN.

.

ABSTRACT

This research aimed to propose a technological framework using the MEAN stack applied to queue management in the Cooperativa de Ahorro y Crédito Tulcán Ltda. The study employed a mixed approach with descriptive and field-based characteristics. The current queue management processes were described in ten savings and credit cooperatives, including an in-depth analysis of the Cooperativa de Ahorro y Crédito Tulcán Ltda. Additionally, the operational impact of these systems was evaluated through indicators such as waiting times and service efficiency. The collected data was processed using tools such as Excel, applying descriptive and correlational statistical techniques. A system design based on the MEAN stack was developed. Finally, a comprehensive technological framework for queue management was formulated, grounded in the MEAN stack: MongoDB, Express.js, Angular, and Node.js, complemented by Socket.io for real-time communication. The study identified that traditional queue management methods generated operational inefficiencies, prolonged waiting times, and reduced customer satisfaction. Through an appropriate methodology and the use of Scrum for development, a web system was implemented that includes in-person and online queue assignment with OTP validation, module transfer, real-time notifications via Socket.io, as well as report and QR-coded ticket generation. The proposed architecture demonstrated advantages in scalability and maintainability by using JavaScript across the entire technological stack. The implementation was validated with end users, successfully addressing the identified deficiencies and establishing a foundation for future technological developments within the cooperative. The MEAN stack technological framework is positioned as an effective solution for modernizing processes in financial institutions.

Keywords: software architecture, cooperatives, digitalization, queue management, MEAN stack.

CAPÍTULO I

PROBLEMA

1.1. PLANTEAMIENTO DEL PROBLEMA

Hoy en día, la automatización y digitalización están cambiando radicalmente el funcionamiento de las instituciones financieras ecuatorianas, permitiendo una generalidad de la propia red de los bancos, promoviendo la eficiencia del sistema operativo, una economía de servicios, y una mayor inclusión financiera, especialmente en los lugares rurales (Giler Araujo et al., 2024). No obstante, existen muchas cooperativas de ahorro y crédito que todavía utilizan métodos tradicionales para la gestión de turnos, lo que las deja con un nivel muy limitado de eficiencia operativa por el lado de la administración del equipo técnico y del servicio al cliente.

Como parte esencial de la economía popular y solidaria, las cooperativas de ahorro y crédito son agentes clave para la inclusión financiera, sin embargo existe una baja satisfacción de servicio debido a la falta de soluciones tecnológicas adecuadas en la administración de turnos y en la evaluación de la satisfacción del cliente, lo que puede conllevar a tiempos de espera extensos, deficiencia de organización en la administración de la atención y en la recopilación de datos necesarios para la mejora del servicio. Siendo así, esta circunstancia refleja las perspectivas de los clientes, con la posible consecuencia de afectar el crecimiento sostenible y la competitividad de las cooperativas de ahorro y crédito.

La Cooperativa de Ahorro y Crédito Tulcán Ltda., situada en la provincia de Carchi, Ecuador, representa un ejemplo específico de esta problemática. En la actualidad, esta cooperativa carece de un software integral de administración de turnos y de eficiencia del servicio, impactando negativamente en la experiencia de usuario y la eficacia operacional. Según Rodríguez (2019), la falta de herramientas tecnológicas adecuadas puede dar una percepción negativa de los usuarios en la cooperativa y limitar su crecimiento.

Si no existe un estudio pormenorizado sobre el tema de las deficiencias y sus efectos en las cooperativas, es posible que se llegue a perder la fidelización y

satisfacción de los usuarios. En un escenario como este, la falta de modernización de la gestión del servicio puede llevar a que la cooperativa sufra una merma en la competitividad frente a otras instituciones financieras que tengan un mejor proceso tecnológico.

Por lo cual se hace notar la necesidad de realizar un estudio que acompañe las diferentes fases de investigación, para identificar y analizar las debilidades en la gestión de la distribución de los turnos de los clientes de la Cooperativa de Ahorro y Crédito Tulcán Ltda., el cual constituyó la base para poder proponer una solución tecnológica que optimice los procesos internos y con ello contribuya a mejorar la eficiencia operativa y por ende la experiencia del servicio de sus clientes.

1.2. PREGUNTAS DE INVESTIGACIÓN O HIPÓTESIS

- ¿Cuáles son los procesos actuales de gestión de turnos utilizados en las cooperativas de ahorro y crédito ecuatorianas?
- ¿Qué características y funcionalidades específicas debe incluir el sistema de gestión de turnos para optimizar la eficiencia operativa y reducir los costos en las cooperativas de ahorro y crédito ecuatorianas?
- ¿Cuál es el impacto potencial del uso del stack MEAN en el desarrollo e implementación del marco tecnológico para la gestión de turnos en las cooperativas de ahorro y crédito ecuatorianas?

1.3. OBJETIVOS DE INVESTIGACIÓN

1.3.1. Objetivo General

Proponer un marco tecnológico con el uso del stack MEAN aplicado a la gestión de turnos en la Cooperativa de Ahorro y Crédito Tulcán Ltda.

1.3.2. Objetivos Específicos

- Identificar el proceso para la gestión de turnos en la Cooperativa de Ahorro y Crédito Tulcán Ltda.
- Diseñar un sistema de gestión de turnos que permita a los clientes tomar turnos de manera eficiente, reduciendo los costos operativos para la Cooperativa de Ahorro y Crédito Tulcán Ltda.

- Desarrollar un marco tecnológico con el uso de stack MEAN aplicado a la gestión de turnos en la Cooperativa de Ahorro y Crédito Tulcán Ltda.

1.4. JUSTIFICACIÓN

Este trabajo tiene como finalidad sustentar empíricamente la necesidad de adoptar una solución tecnológica basada en el stack MEAN para el control de turnos en cooperativas de ahorro y crédito ecuatorianas. La evaluación de la situación actual se realiza a partir de la utilización de indicadores que reflejan la existencia de deficiencias en la infraestructura digital y se argumenta la importancia de la modernización de estos procesos para obtener una mejora en la eficiencia de sus operaciones y la experiencia que tiene el usuario. En este sentido, la implementación de un sistema apoyado en el uso del stack MEAN se presenta como una opción idónea que, si se hace a tiempo, responde a las problemáticas de adaptaciones tecnológicas rápidas y reducción de carga operativa, lo que implica reducción de costos.

Existen datos empíricos que demuestran que la utilización de infraestructuras tradicionales ha generado cuellos de botella en la administración de turnos, repercutiendo negativamente en indicadores clave como los tiempos de espera y errores en los sistemas. Como menciona Sánchez Aristi et al. (2023), "La digitalización del sector financiero y el bancario como parte de este, son una realidad sin discusión. El efecto transformador de las tecnologías exponenciales y la velocidad del cambio que las mismas generan, induce a las entidades a redefinir sus modelos de negocio, sus procesos y la relación con sus clientes, no ya sólo desde la eficiencia, sino también desde la propuesta de valor y la experiencia" (p. 8). Además de estas ventajas, la integración de un único lenguaje de desarrollo, como en el caso del stack MEAN, facilita la comunicación entre componentes y mejora el mantenimiento del sistema y desarrollo evolutivo, dando una mejor cabida a esa nueva relación con el cliente (Agarwal et al., 2024). El resultado final se traduce, pues, no sólo en el hecho de que la solución tecnológica que se propone resuelve los problemas estructurales que arrastramos desde hace tiempo, sino que, además reduce dicha carga asociada a operaciones manuales redundantes y permite responder ágilmente a las necesidades de los usuarios.

El stack MEAN aglutina soluciones tecnológicas integrales desde la gestión de los datos hasta la interfaz de usuario misma, deconstruyendo de este modo un marco conversacional robusto, que soporte las necesidades que se requieren hoy, aquí y también en el futuro (MongoDB Inc., s. f.). Como ejemplo de su pertinencia, MongoDB Inc. (s. f.) afirma que “MEAN.JS es una nueva alternativa moderna full-stack, basada en JavaScript, que permite producir desarrollos web listos para producción, con la simple capacidad de depuración y mantenimiento, que usa MongoDB, los frameworks Express, AngularJS y también Node.js. Esta herramienta permite no solo acelerar el diseño inicial, sino también reducir esfuerzos repetidos y evitar vulnerabilidades típicas, al tiempo que otorga una estructura coherente en las aplicaciones” (párr. 1). Este enfoque tecnológico, agrupando todos los elementos en un mismo entorno de desarrollo, hace que la iteración sin pausa y nuevas funcionalidades que encajen en los cambios del mundo cooperativo pueden evolucionar.

El requerimiento de establecer un sistema que ayude a evitar la complejidad de iniciar la relación con el cliente se basa en el estudio realizado sobre casos prácticos en entornos cooperativos. Un claro ejemplo de esto es la investigación de (Quezada Torres y Chamba Mendez, 2023), la cual analiza la implementación de un sistema CRM en cooperativas de ahorro y crédito en Ecuador, en ella se concluye que la digitalización facilita la gestión de atención al usuario y la eficiencia operativa. Utilizando indicadores como el decremento de errores en el registro de turnos y la disminución en el tiempo de espera, permite medir la magnitud de las carencias de la infraestructura actual, lo que puede justificar el paso hacia una solución tecnológica nueva. La visión analítica presentada por los diferentes estudios indica que las inversiones en tecnologías basadas en metodologías ágiles y sistemas integrados se traducen en una mejora efectiva de la eficiencia interna con el beneficio adicional de una disminución de los costos operativos y el incremento a la competitividad en el mercado (Romani Alejo et al, 2023).

A su vez, la validación de la pertinencia de la propuesta se fundamenta en estudios de caso que demuestran el grado de mejora que generan soluciones integradas en contextos similares. Al analizar resultados obtenidos en empresas que han migrado hacia arquitecturas con tecnologías modernas, se comprueba

que la utilización de un framework unificado disminuye la complejidad del mantenimiento y la gestión de datos. De esta manera, queda claro que la transformación digital, sustentada en evidencias empíricas y en el respaldo de estudios previos, permite mejorar procesos y responder ágilmente a los requerimientos del contexto actual de las Cooperativas de Ahorro Crédito. Por otro lado, Agarwal et al. (2024), afirma que “la integración completa que ofrece el stack MEAN al utilizar un único lenguaje para el desarrollo tanto del frontend como del backend simplifica el proceso de prototipado y facilita la incorporación de mejoras tecnológicas de manera ágil" (págs. 1-2), lo que refuerza la viabilidad y pertinencia de la propuesta presentada para abordar de manera integral las deficiencias identificadas en el sistema de turnos.

En este sentido, la implementación del stack MEAN en la Cooperativa de Ahorro y Crédito Tulcán Ltda., se justifica no solo por sus ventajas técnicas, sino también por el respaldo y el apoyo institucional para ejecutar este proyecto. Esta cooperativa ha facilitado la implementación o ejecución de esta solución tecnológica, como puede observarse el ANEXO E, donde se da la conformidad para el uso del stack MEAN para poder desarrollar un software de gestión de turnos. La finalidad de este software es la optimización de la gestión de turnos, la disminución de tiempo de espera, así como la optimización del rendimiento operativo en las sucursales. La implementación incluirá el uso de MongoDB para el almacenamiento de información, Express.js para la creación de APIs, Angular para la UI y Node.js para la ejecución del backend, integrando todas estas tecnologías en una solución unificada y eficiente.

La implementación del stack MEAN en la Cooperativa de Ahorro y Crédito Tulcán Ltda., no sólo abordaría las carencias existentes en la gestión de turnos, sino que también sentaría bases sólidas en arquitectura y lenguaje para futuros desarrollos de software. Esta solución tecnológica mejoraría la eficiencia operativa, la satisfacción del usuario y la competitividad de la entidad en el sector financiero popular y solidario, demostrando que la adopción de tecnologías modernas permite en gran medida la mejora de procesos.

CAPÍTULO II

FUNDAMENTACIÓN TEÓRICA

2.1. ANTECEDENTES DE INVESTIGACIÓN

Es importante tratar los antecedentes metodológicos de esta investigación, citando investigaciones pasadas que han utilizado o se han basado en el stack MEAN (MongoDB, Express.js, Angular y Node.js) alrededor del mundo y dentro del país que en su desarrollo se han evaluado las características que ofrece y los ambientes en los que han sido utilizado bajo un contexto específico, a continuación, se describen las investigaciones realizadas relevantes al tema tratado desarrolladas en algunos países.

(Pratyusha et al., 2023) en su estudio “Development of a college placement web site using the MEAN Stack” realizado en la India enfatiza la relevancia de desarrollar un sitio en la web con la tecnología stack MEAN para la ubicación de estudiantes dentro del campo laboral. Destaca en él la sinergia entre las tecnologías que lo componen para optimizar la eficiencia, escalabilidad y experiencia del cliente en el sitio web. Este artículo pone de manifiesto la manera en que el stack MEAN unifica la información, intensificándola y también el proceso de ubicación estudiantil y administrativa de instituciones educativas, así como también lo es para los estudiantes a la hora de dar sus primeros pasos en el ámbito laboral, su página web es escalable y puede atender a un gran número de usuarios que navegan en el mismo instante.

Por otro lado, Sarwar et al. (2023) en su trabajo titulado “An Efficient E-Commerce Web Platform Basedon Deep Integration of MEAN Stack Technologies”, destaca la relevancia de la implementación de una herramienta para comercio electrónico que cumpla con los requerimientos de los clientes actuales, una plataforma que permita llevar a cabo un proceso de compra en línea optimizada. Para ello, se hace uso del framework de desarrollo tecnológico stack MEAN. Existe una redistribución de la interactividad entre compradores y vendedores, brindando a los usuarios que estén inscritos la opción de adquirir productos de forma rápida mediante diversas opciones de pago. La página web cuenta también con métodos de gestión de pedidos para propietarios de los

negocios. La conjunción de estas tecnologías permitió ofrecer la creación de una plataforma e-commerce integral y verticalmente escalable, de forma tal que aporta la experiencia del cliente y simplifica la administración de las operaciones para la comercialización a través de Internet. El campo de la gestión del inventario Sharma et al. (2022), en el artículo que publican en la ciudad de Ghaziabad, del país que se encuentra en el estado de Uttar Pradesh, al norte de la India "Transforming Inventory Management System Using MEAN Stack" en el que se habla de la creación de un sistema de gestión del inventario para la industria retail y los problemas existentes de los métodos tradicionales de gestión del inventario basados en hojas de cálculo. En el que proponen un sistema basado en el MEAN stack, con el deseo de brindar una solución a los inconvenientes existentes, a través de una interfaz de usuario más completa y una base de datos dedicada. Se plantea la metodología y se contempla crear un desarrollo web para administrar inventario y ventas. Donde se explican los hallazgos sobre los beneficios de la tecnología MEAN en el desarrollo de software de gestión de inventarios y ventas. En el que se hace hincapié en los beneficios obtenidos en la experiencia de los desarrolladores, escalabilidad y facilidad para implementar en la nube.

Así también, Rana et al. (2024), dicha investigación, la cual se incluye en el artículo titulado "An Efficient Artificial Intelligence (AI) & Internet of Things (IoT's) Based MEAN Stack Technology Applications", describe la implementación de una plataforma de comercio electrónico clasificada mediante el stack MEAN (Node.js, Express.js, Angular.js y MongoDB) junto a IA (Inteligencia Artificial) e IoT (Internet de las Cosas). En este estudio se subraya la potencialidad del stack MEAN a la hora de desarrollar sistemas web escalables y eficientes, asimismo la integración de tecnologías como MongoDB, Angular, Express.js, y Node.js para una experiencia de usuario fluida. El artículo también hace referencia al uso de modelos de predicciones del precio basados en IA, así como chatbots inteligentes para mejorar la experiencia del usuario.

En referencia al Ecuador, se realizaron búsquedas de investigación que tengan como objetivo la exploración de la utilización del stack MEAN en sus diferentes ámbitos. Como resultado, se obtuvieron resultados provenientes de los últimos trabajos que ya están aplicando dicha tecnología como base en el desarrollo e

implementación de nuevas soluciones con el fin de poder ofrecer nuevas ventajas frente a las tecnologías existentes.

Heredia y Sailema (2018) en su artículo titulado "Comparative Analysis for Web Applications Based on REST Services: MEAN Stack and Java EE Stack" en la provincia de Esmeraldas donde realiza un estudio de la aplicación del stack MEAN como de Java EE a las aplicaciones web basadas en servicios REST. De igual forma, se indica la importancia de la selección de las utilidades adecuadas para el frontend, backend y base de datos basadas en servicios REST. La investigación se basa principalmente en el objetivo de análisis de la aplicación del stack MEAN a los sistemas web basados en servicios REST como una comparación hacia la aplicabilidad del stack Java EE. Se desarrollan dos aplicaciones similares para el análisis de las diferencias en cuanto a: rendimiento, tiempo de carga y respuesta. Se puede concluir, por tanto, que se trata de un modelo bastante aplicable, al presentar una relación detallada de las diferencias entre ambas tecnologías, pero a su vez mostrando las similitudes entre las mismas en un aspecto muy detallado respecto a la selección de la arquitectura adecuada a los sistemas de desarrollo para aplicativos web.

El trabajo "Modelo de Gestión de Proceso Apoyado en la Tecnología Stack MEAN" de Reinoso Coello (2019) realizado en el ámbito local de la ciudad de Ambato, provincia de Tungurahua en la realización de su trabajo previo para el grado de Magíster en Gerencia Informática. Este trabajo se basa en el problema que se presenta en relación con la venta y producción de una empresa local, en el sentido que no existe una estructura orgánica y los tiempos de fabricación no son los adecuados y perjudica el crecimiento y funcionamiento de la empresa. El propósito fundamental es crear un modelo de administración de procesos utilizando la tecnología stack MEAN enfocado en la producción. Para el desarrollo de este trabajo se recurre a un enfoque cuantitativo, de diseño no experimental y se aplica la metodología PDCA como camino hacia una gestión efectiva y mejora continua. Se realiza una aplicación web mediante el stack Mean y utilizando la metodología ágil Scrum, consiguiendo de esta manera una gestión eficiente de procesos de producción y venta, logrando una mejora óptima en la producción que se procesa mediante comparación entre procesos manuales versus procesos automatizados.

Choto Maza et al. (2020) en su trabajo "Desarrollo de una aplicación móvil utilizando el framework MEAN Stack e IONIC: Un estudio de caso en una compañía de transporte" realizado en Riobamba, Chimborazo, detalla la ejecución de una app móvil informativa para la gestión de rutas, fundamentada en el marco MEAN Stack e IONIC. Se utilizó la metodología ágil SCRUM en el transcurso del desarrollo. Los creadores evaluaron la aplicación en una prueba de usabilidad bajo una norma ISO, considerando aspectos como comprensibilidad, aprendizaje, funcionalidad, prevención de errores, diseño y accesibilidad. Se llevaron a cabo encuestas cuyos resultados mostraron que una app móvil y web era sencilla de utilizar, aprender y permitía al usuario rastrear la menor ruta. Estos procedimientos contribuyeron a optimizar la gestión informativa en la empresa.

El análisis de estos antecedentes demuestra que el stack MEAN es ampliamente utilizado en el entorno web en diferentes partes del mundo, aunque este también puede ser utilizado para desarrollo de aplicaciones en otros entornos como son las aplicaciones móviles, solamente se registra una investigación que cita el stack MEAN en este entorno, por otro lado las investigaciones se han realizado en muchos ambientes específicos como el transporte hasta el Internet de las cosas pero no se han registrado investigaciones en el ambiente financiero o cooperativas de ahorro y crédito, los estudios revisados coinciden en que la arquitectura del stack MEAN brinda ventajas en aspectos de escalabilidad, eficiencia y mejora de procesos, consolidándose como una alternativa viable para la modernización tecnológica en diversos ámbitos como lo es el financiero.

2.2. MARCO TEÓRICO

2.2.1. Situación actual de la gestión de turnos en las instituciones financieras ecuatorianas

Métodos tradicionales (turnos físicos, llamadas telefónicas).

La administración de turnos en las instituciones financieras de Ecuador todavía representa un desafío a medida que se emplean prácticas tradicionales como el sistema de traspaso de turnos físicos y la utilización del teléfono para concertar turnos. Dichas prácticas, si bien son eficaces a corto plazo, van a ser limitadas por la demanda de agilidad y eficacia que tiene lugar en el entorno digital. Un

estudio reciente expresa que la adopción de funcionalidades como el Big Data y la inteligencia artificial ha permitido a las instituciones financieras mejorar sus prácticas de gestionar riesgos y su campo operativo superando el uso de métodos convencionales de explotación (Olaiya et al., 2024). No obstante, muchas instituciones en Ecuador están todavía en desuso durante el tiempo, lo que puede limitar la rápida adaptación a un entorno financiero que va cambiando rápidamente. La adopción de sistemas actuales y no tradicionales para las instituciones financieras ha pasado a ser una necesidad urgente; sobre todo tras la llegada de la COVID-19 que ha obligado a adoptar un enfoque más ágil y tecnológico en procedimientos operacionales (Saeed et al., 2022). Este paso permitirá no solamente una mejora en la operación, sino que también en el cuidado del cliente y en la sistematización de recursos.

2.2.1. Sistemas tecnológicos existentes (web, aplicaciones móviles).

La realidad de la gestión de los turnos en las instituciones financieras del Ecuador evidencia la creciente incorporación de sistemas tecnológicos como aplicaciones móviles o plataformas web que permiten la optimización de la experiencia del cliente y la eficiencia de la operación (Gimba et al., 2020). Los usuarios pueden desde diferentes dispositivos, gestionar sus turnos anticipadamente, cumplir con ellos, reducir el tiempo de espera en las sucursales; reduciendo la demanda de servicios, y mejorando así la distribución (Safdar et al., 2021).

Un reciente estudio indica que el uso de aplicaciones móviles mejoró la gestión de las citas, logrando un alto nivel de satisfacción del cliente, debido a la ampliación del rango de opciones y más comodidad (Gimba et al., 2020). Además, las plataformas web son las que dieron lugar a consolidar el sistema; ya que permiten que el cliente gestione los turnos desde sus dispositivos y por una conexión a internet que es especialmente importante en un entorno después de una pandemia que se impuso por las medidas de digitalización (da Rocha Nascimento, 2021).

Pero a pesar de estos avances se percibe que aún existen "desafíos" sobre la brecha tecnológica y la llegada oficiosa de estos sistemas en algunos lugares del país que limitan su popularidad (Safdar et al., 2021). En general, la adopción

de sistemas tecnológicos para la gestión de los turnos revela un alto avance de las instituciones financieras del Ecuador, pero tiene que ser un enfoque inclusivo que maximice los beneficios de dicho sistema.

La Tabla 1 presenta una comparación factible que ayuda a observar las principales características entre los métodos tradicionales y los sistemas tecnológicos; y muestra la comparación de estos métodos a modo de mostrar las diferencias fundamentales (por ejemplo, la Accesibilidad y Flexibilidad); además ayuda a entender las principales características de los métodos tradicionales frente a los tecnológicos, que son importantes para la comprensión del estudio realizado.

Tabla 1

Métodos tradicionales vs sistemas tecnológicos actuales

Criterio	Métodos Tradicionales (Turnos Físicos, Llamadas Telefónicas)	Sistemas Tecnológicos Existentes (Web, Aplicaciones Móviles)
Accesibilidad	Restringida a la disponibilidad física o a la línea telefónica en horario laboral.	Disponible en todo momento y cualquier lugar con acceso a internet.
Comodidad para el usuario	Requiere desplazamiento físico o llamadas, lo cual puede ser inconveniente.	Permite agendar turnos desde casa o cualquier ubicación sin desplazarse.
Tiempo de espera	Tiempo de espera generalmente más largo debido a la falta de optimización.	Reduce el tiempo de espera mediante la asignación de turnos programados.
Flexibilidad	Limitada, ya que los turnos se deben solicitar en persona o por llamada.	Alta flexibilidad; los usuarios pueden elegir y modificar turnos según su conveniencia.
Precisión en la asignación de turnos	Sujeto a fallos humanos en la asignación y administración de turnos.	Alta precisión mediante sistemas automatizados que minimizan errores.

Criterio	Métodos Tradicionales (Turnos Físicos, Llamadas Telefónicas)	Sistemas Tecnológicos Existentes (Web, Aplicaciones Móviles)
Costo operativo	Costos elevados relacionados con la necesidad de personal para gestionar turnos.	Costos operativos reducidos; mayor inversión inicial en tecnología, pero menos gasto en personal.
Capacidad de manejo de volúmenes	Limitada capacidad para manejar grandes volúmenes de solicitudes simultáneas.	Capacidad para gestionar un gran número de solicitudes de manera eficiente.
Trazabilidad y análisis de datos	Difícil de implementar; registros manuales o limitados.	Permite el análisis y seguimiento de datos en tiempo real, propiciando mejoras continuas.
Impacto ambiental	Requiere uso de papel para registros físicos.	Reduce el uso de papel al digitalizar la gestión de turnos.
Satisfacción del cliente	Generalmente más baja debido a los inconvenientes mencionados.	Usualmente mayor por la comodidad y la minimización de tiempos de espera.

2.2.2. Gestión de turnos en las instituciones financieras

Impacto en la experiencia del cliente.

La gestión eficiente de turnos en las instituciones financieras permite mejorar la operación interna y la experiencia del cliente. Un sistema de gestión de turnos bien implementado reduce los tiempos de espera, mejora la satisfacción del cliente y aumenta la productividad del personal (Basantes Espinoza, 2020).

La introducción de tecnologías avanzadas para gestionar las colas, como el uso de aplicaciones móviles y sistemas de reservas vía Internet, ha permitido a las entidades de crédito adaptarse mejor a las necesidades del cliente contemporáneo, quien prioriza la eficiencia y la conveniencia (López y Martínez, 2023). La introducción de tecnologías avanzadas para gestionar las colas, como el uso de aplicaciones móviles y sistemas de reservas vía Internet, ha permitido

a las entidades de crédito adaptarse mejor a las necesidades del cliente, quien prioriza la eficiencia y la conveniencia (Vera León, 2023).

Las mejoras en la experiencia de los clientes no solo ayudan a generar vínculos, sino que también ayudan a que se establezca una buena imagen de marca y generar una mejor rentabilidad a largo plazo para aquellas entidades financieras (Méndez, 2023). Para concluir, la gestión de turnos no debe ser vista exclusivamente como un sistema operativo sino como un sistema estratégico que tiene repercusiones en la satisfacción de los clientes y también en el éxito competitivo de las entidades.

2.2.3. Metodologías de desarrollo de software

Una metodología implica principios, procedimientos y prácticas que surgen de la necesidad de crear un producto específico y lograrlo con alta probabilidad de éxito hasta finalizar. Este marco de trabajo es crucial para el desarrollo de sistemas, ya que enfatiza las consideraciones técnicas, organizativas, de proyecto y de equipo (Martínez et al., 2022).

Metodologías tradicionales de desarrollo.

Las metodologías tradicionales de desarrollo se definen por un proceso organizado y disciplinado, y fueron creadas para identificar eficazmente los requerimientos del cliente y producir software de calidad. Estas metodologías ven el proyecto como uno de gran tamaño, con requisitos estables que necesitan ser organizados y documentados, sus requerimientos se pactan una vez para todo el proyecto. (Abuchar Porras, 2023)

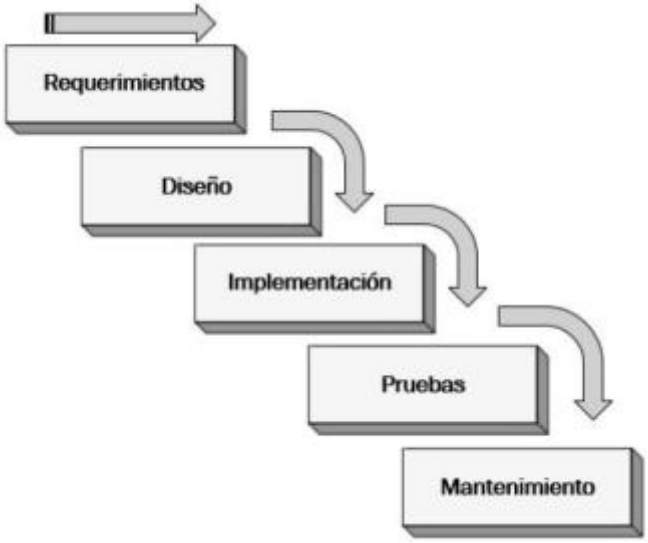
Según Morales Carrillo et al. (2022) las metodologías convencionales definen todos los requerimientos al inicio del proyecto, lo que las vuelve inflexibles. Se centran en un control de procesos, determinando las actividades, el alcance, las herramientas necesarias y el seguimiento, es decir, se efectúa una planificación completa del trabajo a realizar. Estas metodologías se definen por no ajustarse bien a los cambios, lo que genera altos costes al implementarlas.

El modelo en cascada es una metodología tradicional que plantea un enfoque sistemático y secuencial para desarrollar software, centrado en el análisis, diseño, implementación, pruebas y mantenimiento; al finalizar cada una de las

etapas busca generar documentación para garantizar que cumple con las especificaciones solicitadas. Al pasar el tiempo se logra detectar los principales problemas de este método pues su rigidez no permite responder a los requerimientos cambiantes del cliente, además que no es un modelo apto para proyectos de gran tamaño. (Aguirre Barrera y Aguirre Barrera, 2020). En la Figura 1 se visualiza un diagrama de la metodología de Ciclo de vida o Cascada.

Figura 1

Diagrama de metodología en Cascada



Nota. Tomado de: (Aguirre Barrera & Aguirre Barrera, 2020)

En conclusión, el uso de esta metodología tradicional no es efectiva dado que es un proyecto de gran magnitud, por lo que tiene sentido la entrega gradual y a su vez se necesita avanzar de forma rápida; se considera que una metodología ágil es más adaptable a las necesidades del proyecto.

Metodologías de desarrollo ágil.

Las metodologías ágiles destacan por su flexibilidad, ya que priorizan la adaptabilidad sobre el cumplimiento riguroso de un plan. Este enfoque se lleva a cabo de manera integral o por etapas, permitiendo dividir el proyecto en partes más pequeñas, de este modo, se pueden entregar las funciones operativas del software rápidamente, comenzando por los elementos más críticos del negocio y aumentando la satisfacción del cliente (Morales et al., 2022).

Las metodologías ágiles tales como Scrum, Extreme Programming (XP) y Kanban están claramente caracterizadas por un enfoque flexible y adaptable que permite dar respuesta a los cambios producidos en las solicitudes del cliente con el fomento del desarrollo iterativo e incremental de software funcional. En la región 7 del Ecuador, estas metodologías también han mostrado efectividad en la mejora de la colaboración de los equipos, la reducción de los tiempos de entrega y el aumento de la calidad del software a pesar de que su implementación se encuentra con un camino plagado de retos como el de la falta de formación y la resistencia al cambio (Armijos et al.,2025). Tal y como se puede visualizar en la Figura 2 del estudio, el crecimiento de la adopción y evolución de las metodologías ágiles desde que estas nacieron en el 2001 hasta la adopción de éstas en Ecuador suponen una curva de crecimiento muy significativa a partir de 2013, año en que empezaron a ser adoptadas en los proyectos de la región, lo cual denota la pertenencia de estas en el sector del desarrollo del software en la región.

Figura 2

Evolución de metodologías ágiles en Ecuador

Nota. Tomado de: (Armijos Ortega, Vélez Macas, & Lojan Cueva, 2025).

Metodologías tradicionales vs metodologías ágiles.

Las metodologías de desarrollo tradicionales se caracterizan por una estricta disciplina secuencial realizando un esfuerzo considerable en la planificación global y completa del proyecto, y solo una vez que esta planificación está detallada, se inicia el ciclo de desarrollo. En cambio, las metodologías ágiles

priorizan la ejecución del trabajo en sí, realizando procesos pequeños a corto plazo, buscando un equilibrio entre el proceso y el esfuerzo, permitiendo una mayor retroalimentación continua en el proyecto (Romero et al., 2022).
La Tabla 2 muestra las características relevantes entre estos dos grupos de metodologías.

Tabla 2

Comparación entre Metodología Tradicional y Ágil

Metodologías Tradicionales	Metodologías Ágiles
Proceso rígido	Proceso flexible
Predictivos	Adaptativos
Resistencia a los cambios	Fácil adaptación a los cambios durante el proyecto
La arquitectura del software es fundamental y se representa a través de modelos	La arquitectura del software se considera menos relevante
Se concibe como un solo proyecto	Un proyecto se divide en varios proyectos más pequeños
Orientado a procesos	Orientado a personas
Basadas en reglas derivadas de los estándares del entorno de desarrollo	Basadas en heurísticas derivadas de las prácticas de producción de código
Documentación extensa	Poca documentación

Nota. Adaptado de Canós, Letelier, & Penadés (2021).

Metodología de desarrollo SCRUM

Scrum es un marco de trabajo ágil que busca optimizar la colaboración efectiva dentro de un equipo de trabajo, esto significa que proporciona una estructura flexible que se puede adaptar a las necesidades específicas de cada proyecto y equipo. Ofrece un conjunto de principios, prácticas y roles, pero no prescribe una

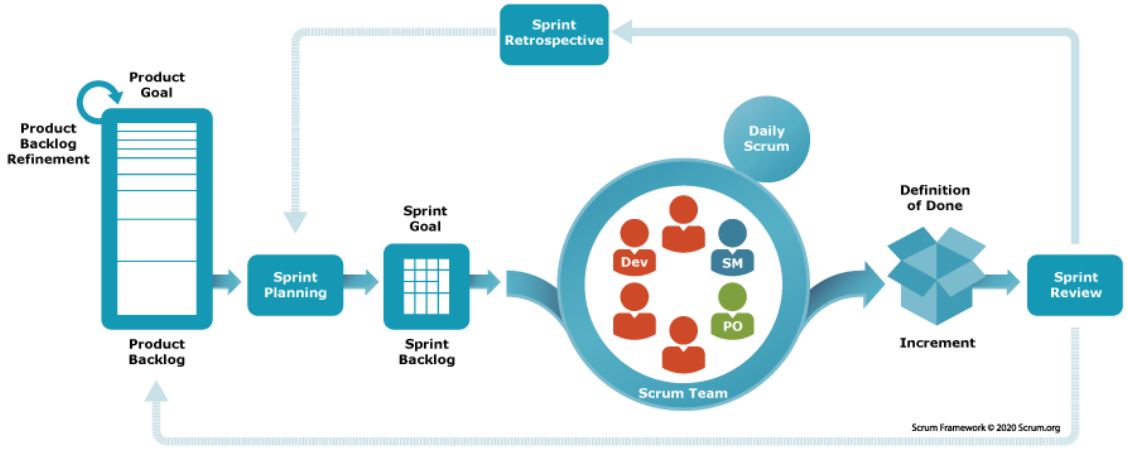
forma rígida de hacer las cosas, facilitando la gestión y la ejecución del trabajo de manera eficiente (Schwaber y Sutherland, 2020).

Según Canós, et al. (2021) SCRUM establece un marco para gestionar proyectos, especialmente los que requieren cambios rápidos. Sus características fundamentales se resumen en dos: el desarrollo se lleva a cabo a través de interacciones de hasta 30 días, llamadas Sprints. Cada sprint produce un ejecutable del proyecto que se indica al cliente. Otro aspecto clave son las reuniones del equipo durante el proyecto que facilitan su coordinación e integración.

El marco de trabajo Scrum incluye al equipo Scrum, eventos, reglas y artefactos relacionados. Cada elemento tiene un rol particular y es vital para el éxito de Scrum; en la Figura 3 se puede visualizar cómo se gestiona típicamente esta metodología.

Figura 3

Visión general de SCRUM



Nota. Tomado de: (Chavez Ponce et al., 2022)

Equipo de Scrum. La unidad fundamental de Scrum es el Equipo Scrum (Scrum Team), consiste en un Propietario del Producto (Product Owner), el Equipo de Desarrollo (Development Team) y un Scrum Máster; este equipo se centra en la autoorganización y la multifuncionalidad para optimizar la flexibilidad, la creatividad y la productividad (Schwaber y Sutherland, 2020). En la Tabla 3 se detallan los roles del equipo en la metodología Scrum.

Tabla 3

Roles del equipo Scrum

Rol	Definición
Product Owner	Es el representante de los accionistas y usuarios del software, por lo que es el único responsable de administrar el Producto (Product Backlog). Transmite la visión del producto al equipo, gestionando y priorizando los requisitos.
Development Team	Son los encargados de convertir la lista de requerimientos en funcionalidades del software. Comparten un propósito y asumen juntos la responsabilidad del trabajo realizado.
Scrum Master	Actúa como enlace entre el dueño del producto y el grupo de desarrollo, asegurando la comprensión y la aplicación correcta de Scrum. Su rol es fomentar la creatividad y la productividad del equipo, y hacer visibles sus avances al dueño del producto.

Eventos de SCRUM. Scrum incorpora una serie de eventos programados con el objetivo de generar regularidad en el trabajo del equipo y minimizar la necesidad de reuniones improvisadas. Cada uno de los eventos de Scrum constituye una oportunidad formal para la inspección y adaptación en algún aspecto (Schwaber y Sutherland, 2020). En la Tabla 4 se detallan los eventos de Scrum principales.

Tabla 4

Eventos de Scrum

Eventos	Descripción	Duración
Sprint	Periodo en el que se realiza un desarrollo incremental del producto "Terminado", utilizable y potencialmente desplegable.	Es un período de tiempo fijo de un mes como máximo.

Eventos	Descripción	Duración
Sprint Planning	En la reunión de preparación del Sprint, el equipo Scrum colabora en conjunto para crear el plan de trabajo que se ejecutará durante el Sprint.	Un Sprint de un mes tiene una duración máxima de ocho horas.
Daily Scrum	El Scrum diario es una junta diaria en el Sprint, donde el equipo de desarrollo coordina su trabajo y organiza actividades para las próximas 24 horas.	Es una cita con duración de 15 minutos.
Sprint Review	Esta junta se realiza al fin de cada sprint con la finalidad de examinar el desarrollo incremental creado y, de ser necesario, ajustar la pila del producto, de manera que se corrijan estas falencias para el próximo sprint.	Es una junta con un tiempo no mayor a cuatro horas para Sprints mensuales.
Sprint Retrospective	El propósito de esta reunión es analizar detalladamente el trabajo realizado durante el sprint, identificando los aspectos positivos y las áreas de mejora en el proceso.	Es una junta limitada no mayor a tres horas para Sprints mensuales.

Artefactos de SCRUM. Para Scrum, los artefactos son cualquier elemento tangible que representa el trabajo ejecutado o el valor que se ha generado en el proceso de desarrollo. Su propósito principal es aumentar la transparencia, ayudando a la inspección del progreso y a adaptarse a los cambios. El diseño de estos artefactos se centra en la claridad de la información importante, asegurando que todos los miembros del equipo y los interesados tengan una comprensión uniforme de la misma. (Schwaber y Sutherland, 2020). En la Tabla 5 se enumeran los principales artefactos de Scrum.

Tabla 5

Artefactos de Scrum

Artefactos	Descripción
Product Backlog	Es un listado estructurado de lo esencial para el producto, como única fuente de consulta de requisitos ante cualquier modificación.
Sprint Backlog	Combina elementos elegidos del backlog del producto con un esquema para entregar la mejora y lograr la meta del sprint.
Incremento	Es la suma del trabajo completado en el Sprint actual más el de los Sprints anteriores, listo para ser utilizado y cumpliendo la definición de "Terminado".

Beneficios de Scrum. La utilización de Scrum en el transcurso del desarrollo del proyecto de control de turnos para las entidades financieras puede contribuir a generar los siguientes beneficios:

- **Satisfacción del cliente:** la aproximación Scrum incluye al cliente como un sujeto activo del propio control, lo que permite la comprobación del avance del producto que se va implementando como también del proceso en su conjunto. Esta intervención del cliente es importante para contribuir al control del engranaje del equipo y para asegurar que el producto final satisfaga las especificaciones.
- **Mayor eficiencia:** Scrum promueve la plena utilización de los recursos financieros y de tiempo existentes.
- **Mejora de la calidad del producto:** comprobación de los productos obtenidos y que las especificaciones del cliente mejorando el cumplimiento de este.
- **Mejora en la comunicación y colaboración del equipo:** Scrum estimula la colaboración de las personas implicadas. Por tal motivo el resultado es un equipo que evoluciona continuamente, aprendiendo y adaptándose en cada iteración de modo que pueda mejorar su rendimiento.

Selección técnica de metodología

Con base en los conceptos previamente expuestos, se presenta a continuación una comparativa técnica que se puede visualizar en la Tabla 6 y un análisis de la selección de Scrum como metodología de desarrollo para el sistema de gestión de turnos de la Cooperativa de Ahorro y Crédito Tulcán Ltda., considerando las exigencias y requerimientos del proyecto, así como las características propias de cada enfoque de desarrollo de software.

Dado que el proyecto requiere entregas sucesivas, la capacidad de adaptarse a cambios institucionales, la validación temprana de funcionalidades y una interacción continua con los usuarios finales, Scrum se posiciona como el marco metodológico más adecuado en comparación con las metodologías tradicionales (Pressman & Maxim, 2020; Schwaber & Sutherland, 2020).

Se concluye que la aplicación de Scrum contribuye a reducir el riesgo técnico, facilita la detección temprana de errores, promueve una arquitectura de software evolutiva y permite alcanzar una mayor calidad del producto mediante un desarrollo iterativo, la colaboración continua y la mejora constante, características clave en el contexto del presente proyecto y en coherencia con los objetivos de esta investigación.

Tabla 6

Comparativa técnica de metodologías de desarrollo

Criterio técnico	Metodologías tradicionales	Metodología ágil (Scrum)
Gestión de cambios en requisitos	Poca flexibilidad; cambios requieren replanificación completa.	Alta flexibilidad; cambios gestionados en el backlog y sprints.
Entrega de funcionalidades	Entrega única al final del proyecto.	Entregas incrementales frecuentes.
Retroalimentación del usuario	Retroalimentación tardía, casi al final.	Retroalimentación continua en cada iteración.
Control de calidad del software	Pruebas al final del desarrollo.	Pruebas continuas e integración constante.

Criterio técnico	Metodologías tradicionales	Metodología ágil (Scrum)
Visibilidad del avance del proyecto	Avance poco visible hasta etapas finales.	Alta visibilidad mediante reuniones diarias y revisiones de sprint.

Nota. Adaptado de Pressman y Maxim (2020).

Resumiendo, el contenido presentado. Scrum es un marco ágil que contribuye a cada equipo a brindar resultados de calidad rápidamente, eficientemente y de forma flexible, incrementando la satisfacción a los clientes y permitiendo mejoras en la colaboración y la transparencia del equipo. Su característica de ser un proceso iterativo e incremental sumando los roles, los eventos y los artefactos que introduce lo consideran como una técnica potente de gestión para el desarrollo de proyectos complejos en entornos difíciles.

2.2.4. El stack MEAN como solución tecnológica para el desarrollo web

Cada proyecto de software tiene sus características específicas y requiere una determinada combinación de tecnologías, ya que no existe una definición rígida de cuál sería el mejor stack tecnológico. La elección requiere examinar con cuidado los requerimientos del proyecto, así como las destrezas del grupo de desarrollo. La elección del lenguaje de programación o del framework frontend tiene que depender de una estrategia determinada manteniendo en cuenta la escalabilidad, la mantenibilidad y la comunidad de desarrolladores. Un buen marco de trabajo debe dar al equipo las herramientas necesarias para construir aplicaciones de calidad y lo debe hacer de forma eficiente (Weber, 2022).

JavaScript ha transformado el desarrollo web permanentemente, impulsando la creación de aplicaciones completas con un solo lenguaje. MEAN, MERN, MEVN y MEEN son solo algunos de los stack que hay actualmente. Todos incluyen la base de datos MongoDB para gestionar datos, el framework de backend Express.js, el entorno Node.js y un framework frontend específico para ofrecer la interfaz de usuario. Otros stack tecnológicos muy conocidos que no están basadas en JavaScript son los stack LAMP (Linux, Apache, MySQL, PHP) o

LEMP (Linux, Nginx, MySQL, PHP) que tienen a PHP como la tecnología de backend y Linux como el entorno de desarrollo web dominante (Weber, 2022). En la Tabla 7 se presenta un resumen de estos stack con el uso de JavaScript y sin ella.

Tabla 7

Resumen de los stack de tecnología

Full-Stacks con JavaScript	Full-Stacks sin JavaScript
MERN (MongoDB, ExpressJS, ReactJS, NodeJS)	LAMP (Linux, Apache, MySQL, PHP)
MEAN (MongoDB, ExpressJS, AngularJS, NodeJS)	LEMP (Linux, Nginx, MySQL, PHP)
MEVN (MongoDB, ExpressJS, VueJS, NodeJS)	Django stack (Python, Django, MySQL)
MEEN (MongoDB, ExpressJS, EmberJS, NodeJS)	Ruby on Rails (Ruby, SQLite, Rails)

Nota. Adaptado de Weber (2022).

A diferencia de los stack JavaScript que utilizan un solo lenguaje, otras pilas requieren aprender diferentes lenguajes de programación. La elección de un stack tecnológico depende en gran medida de las preferencias y habilidades del desarrollador.

Stack MEAN

El stack MEAN es una opción sólida para crear aplicaciones web usando JavaScript en cada capa. Se compone de cuatro tecnologías clave: MongoDB (base de datos), Express.js (marco para el servidor), AngularJS (marco para el cliente) y Node.js (plataforma para el servidor). Todos estos elementos son creados por equipos distintos y tienen el apoyo de una comunidad activa de programadores (Sayago Heredia y Revelo Bautista, 2022).

Según Agarwal et al. (2024) MEAN.JS es una solución actual de JavaScript fullstack que ayuda a diseñar aplicaciones web de producción muy rápidas, depurables y mantenibles utilizando MongoDB, Express, AngularJS y Node.js. El principal beneficio del stack MEAN es que es muy rápido para crear prototipos debido a que Node.js permite usar JavaScript tanto en el backend como en el frontend, lo que puede ahorrarle la necesidad de aprender un lenguaje separado. Aparte de esto, el comportamiento NoSQL de MongoDB permite modificar y alterar rápidamente la capa de datos sin tener que preocuparse por las migraciones y cambios en las especificaciones.
Se ofrece a continuación una explicación exhaustiva de las herramientas que integran el stack MEAN.

MongoDB

MongoDB es una base de datos NoSQL basada en documentos que ofrece almacenamiento flexible y escalable, gestionando volúmenes significativos de datos no estructurados y semiestructurados, ideal para aplicaciones web modernas y Big Data. MongoDB proporciona alta disponibilidad y escalado automático, asegurando que las aplicaciones manejen crecientes volúmenes de datos sin sacrificar rendimiento ni fiabilidad. Su esquema dinámico permite una fácil modificación y evolución de los modelos de datos a medida que las necesidades empresariales cambian con el tiempo (Le, 2023).
Ngoc Le (2023) asegura que MongoDB es una base de datos de documentos flexible y de código abierto. Utiliza una arquitectura de escalabilidad horizontal, que almacena datos como documentos en formato BSON y admite JSON para su recuperación. La principal ventaja es la escalabilidad, porque facilita el manejo efectivo de vastos conjuntos de datos a través de la cooperación de máquinas menores.

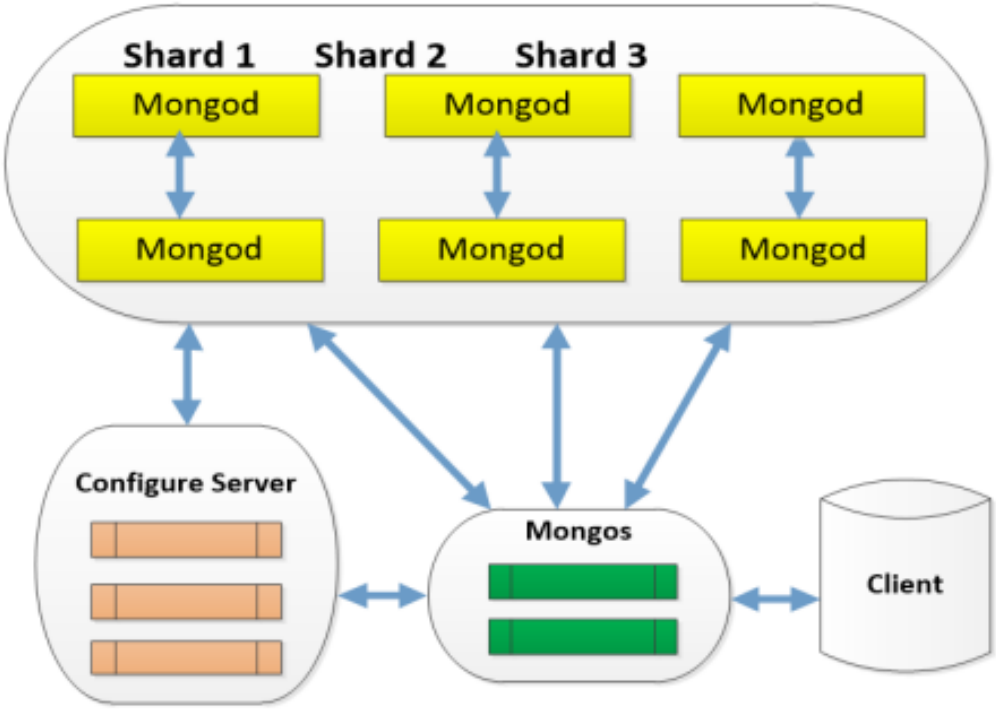
Características. MongoDB es un sistema de gestión de bases de datos NoSQL que permite almacenar datos en documentos con formato JSON, aunque los esquemas son opcionales (Mohanish et al., 2022). Las principales características de MongoDB se enumeran a continuación:

- Consultas ad hoc: MongoDB admite búsquedas por campo, rango y expresiones regulares. Las consultas pueden especificar los campos a devolver de los documentos e integrar funciones JavaScript definidas por el usuario. También se pueden configurar para devolver una muestra aleatoria de resultados de un tamaño determinado.
- Replicación: MongoDB asegura que los datos estén siempre disponibles mediante la creación de conjuntos de réplicas. Estos conjuntos mantienen múltiples copias de la información, lo que permite que la aplicación siga funcionando incluso si uno de los servidores falla. En caso de fallo del servidor principal, otro servidor asume automáticamente su rol, garantizando la continuidad del servicio.
- Balanceo de carga: MongoDB utiliza la fragmentación para distribuir los datos en varios servidores, llamados fragmentos. El usuario define una clave de fragmento, que determina cómo se dividen y distribuyen los datos en diferentes rangos o mediante funciones hash. MongoDB puede ejecutarse en múltiples servidores y ajustar dinámicamente la distribución de datos para ajustarse a los requerimientos de la aplicación.
- Almacenamiento de documentos: MongoDB puede utilizarse como un sistema de archivos, llamado GridFS, con funciones de ajuste de carga y replicación de datos en diferentes máquinas. Esta función, llamada sistema de archivos del sistema, está incluida en los controladores de MongoDB.
- En MongoDB los campos de los documentos pueden ser indexados para optimizar las búsquedas. Se pueden crear diferentes tipos de índices, incluyendo índices primarios y secundarios.
- Utilización en el lado del servidor de JavaScript: MongoDB permite ejecutar código JavaScript directamente en el servidor para realizar consultas, agregaciones de datos y otras operaciones.
- Colecciones con límite: MongoDB ofrece colecciones de tamaño fijo, que mantienen el orden de inserción de los datos y, al llegar al tamaño definido, se comportan como una cola circular.

Arquitectura. MongoDB utiliza una arquitectura distribuida para almacenar y gestionar datos en clústeres. Los datos se dividen en fragmentos replicados y si un servidor falla, las réplicas de ese fragmento aseguran la disponibilidad de los datos. Las operaciones de lectura y escritura se dirigen a fragmentos específicos, y en caso de fallo del servidor principal, uno de los servidores de respaldo asume su rol. Los servidores de configuración gestionan los metadatos y dirigen las solicitudes del cliente a los fragmentos correspondientes. MongoDB utiliza archivos mapeados en memoria e índices de árbol B para optimizar el rendimiento y permite a los usuarios definir claves de fragmento para controlar la distribución de los datos (Khan et al., 2023). La Figura 4 muestra los nodos de fragmentos, los servidores de configuración y los servidores de enrutamiento (o mongos) que conforman la arquitectura de MongoDB.

Figura 4

Arquitectura de MongoDB



Nota: Tomado de: (Khan, Wisal et al., 2023)

Express

Express es un framework liviano del lado del servidor que se ejecuta sobre Node.js. Al ser una aplicación mínima y flexible, es un grupo combinado de utilidades que gestionan la interacción entre el frontend y el almacenamiento de datos, mejorando la transferencia de datos. Si bien proporciona un conjunto de funciones para aplicaciones web, se utiliza para crear API sólidas, administrar solicitudes HTTP y renderizar el enrutamiento (Weber, 2022).

Sayago Heredia y Revelo Bautist (2022) sostienen que Express es simple de configurar, ejecutar, gestionar y ofrecer múltiples elementos esenciales para manejar las solicitudes web. Asimismo, es un marco ligero y versátil que permite que sea más fácil la construcción de aplicaciones web y servidores HTTP básicos. En MEAN, Express actúa como puente que envía las peticiones del cliente a la base de datos y envía cada resultado al cliente.

Ventajas de Express. Express.js es un marco de trabajo para Node.js que minimiza enormemente el desarrollo de aplicativos web. Su diseño minimalista y eficiente permite a los desarrolladores crear aplicaciones web de forma rápida y sencilla, ahorrando tiempo y esfuerzo. Prati y Manjunath (2021) las principales ventajas que destacan del framework son:

- Agiliza y simplifica enormemente la construcción de aplicativos web con Node.js.
- Fácil de configurar y personalizar.
- Permite integrar diversas aplicaciones y servicios de terceros con Express.JS.
- Facilita la definición de rutas de aplicación según métodos HTTP y URL.
- Contiene diversos módulos de middleware que posibilitan ejecutar rutinas.
- Conexión sencilla a bases de datos como MongoDB, Redis y MySQL.
- Permite la generación de servidores API RESTful.

Angular

Angular es un marco de JavaScript empleado para la creación de sitios web. Google lo sostiene como un marco de aplicación web abierto. Se usa principalmente para crear aplicaciones de tipo de una sola página (Single Page

Application, SPA) o que consiste en un único documento HTML. Estas aplicaciones se caracterizan por su capacidad para cargar contenido de manera dinámica, ofreciendo una experiencia de usuario ágil y sensible, equiparable a la de las aplicaciones de escritorio. Su arquitectura se fundamenta en el patrón de diseño Modelo-Vista-Controlador (MVC), lo que facilita la organización y el mantenimiento del código (Gowda, 2022).

MVC es un principio de diseño orientado a objetos que exige la separación del control, lo que significa que una sola función solo puede realizar una cosa. Es la manera convencional de dividir incluso las interfaces de usuario más complicadas en tres perspectivas: lógica de negocio, lógica de interfaz y lógica de entrada. Por ende, el modelo de gestión datos y lógica; la vista presenta la interfaz para el usuario; y el controlador actúa ante las acciones del usuario. El controlador actúa como un puente, traduciendo las interacciones del usuario en acciones que modifican el modelo y actualizan la vista. Esta arquitectura permite realizar cambios de manera eficiente y simplifica la realización de pruebas (Vyas, 2022).

Características. Angular es un gran framework que incluso se le llama plataforma, esto se debe a que angular es compatible con diversas funciones. Esto ayuda a:

- Controlar la interfaz de usuario
- Reaccionar a la entrada del usuario
- Validar la entrada del usuario en formularios
- Enrutamiento y la gestión del estado
- Enviar solicitudes HTTP Ajax
- Proporcionar soporte fuera de línea y capacidades de PWA
- Garantizar pruebas integrales y desarrollo de aplicaciones
- Gestionar múltiples aplicaciones y su conexión

Aunque numerosos frameworks simplifican la construcción de interfaces interactivas, Angular brilla por proporcionar un conjunto integral y sólido de herramientas. Además de facilitar la manipulación eficiente del DOM, ofrece una interfaz de línea de comandos (CLI) oficial que simplifica la creación, gestión,

actualización e implementación de proyectos. Además, Angular promueve la creación de componentes de interfaz de usuario reutilizables, que se pueden combinar para construir aplicaciones complejas (Vyas, 2022).

Ventajas de Angular. De acuerdo con Prati y Manjunath (2021) las ventajas de angular se enumeran a continuación:

- Enlace de datos bidireccional: Angular se creó con una arquitectura Modelo-Vista-Controlador. El framework sincronizó el modelo y la vista. A medida que los datos en el modelo cambian, la vista también lo hace. El enlace de datos bidireccional permitió reducir el tiempo de desarrollo, debido a que no requirió código adicional para la sincronización continua entre el modelo y la vista.
- Compatible con REST: La Transferencia de Estado de Representación permite que la aplicación interactúe rápidamente con el servidor y obtenga los datos necesarios para interactuar con las páginas web.
- Arquitectura de Diseño Mejorada: Angular simplifica la gestión de sus componentes. La arquitectura está diseñada de tal manera que facilita al programador localizar y desarrollar el código.
- Permite vincular de manera sencilla las vistas HTML al modelo lógico del software a través de data binding y otras directivas propias de angular.
- Inyección de Dependencias.
- Soporta el funcionamiento del Modelo-Vista-Controlador.

Node.js

NodeJS es un ambiente multiplataforma y de código abierto para ejecutar JavaScript que ayuda a los desarrolladores elaborar aplicaciones del lado del servidor, cuenta con una extensa biblioteca de JavaScript que facilita el desarrollo web, capacidad para gestionar múltiples solicitudes de clientes al mismo tiempo y soporte para aplicaciones en tiempo real con conexiones bidireccionales. NodeJS se utiliza porque permite a los desarrolladores usar JavaScript tanto en el frontend como en el backend de las aplicaciones web, lo que lo convierte en una herramienta versátil y eficiente para crear aplicaciones completas (Le, 2023).

Ventajas de usar Node JS: De acuerdo con Mohanish et al. (2022) Node fue creado pensando en el desarrollo web, por lo que aborda sus principales desafíos. Entre las principales ventajas se incluyen un rendimiento rápido, actualizaciones periódicas de las herramientas de la biblioteca, una sintaxis flexible y un intercambio de código eficiente.

- Stack tecnológico potente: Node.js se basa en JavaScript, lo que permite el acceso al stack tecnológico de JS, sus recursos y la comunidad. Utilizar Node.js permite crear proyectos con MEAN Stack, que combina MongoDB, Express.js, Angular y Node.js para satisfacer las demandas del desarrollo web:
  - ✓ MongoDB para la gestión de datos;
  - ✓ Express.js para un desarrollo backend eficiente;
  - ✓ Angular para el front-end, ejecutando código JS para crear una interfaz de usuario dinámica;
  - ✓ Node.js para servicios de desarrollo back-end con JavaScript.
- Modelo rápido basado en eventos: Node.js destaca por su rapidez y eficiencia en el manejo de solicitudes, superando a tecnologías como Java, PHP y Go. Las aplicaciones construidas con Node.js pueden responder instantáneamente a las interacciones del usuario.
- Flexible para el desarrollo de microservicios: Al contar con un gestor de paquetes con módulos gratuitos, los desarrolladores pueden editar varios módulos simultáneamente, lo cual es perfecto para la arquitectura de microservicios.
- Ecosistema enriquecido: El gestor de paquetes de Node.js también es una colección de plugins JS de código abierto. NPM recopila más de 840.000 librerías. Los desarrolladores de JS y Node.js utilizan módulos de NPM en más del 97 % de sus aplicaciones web.

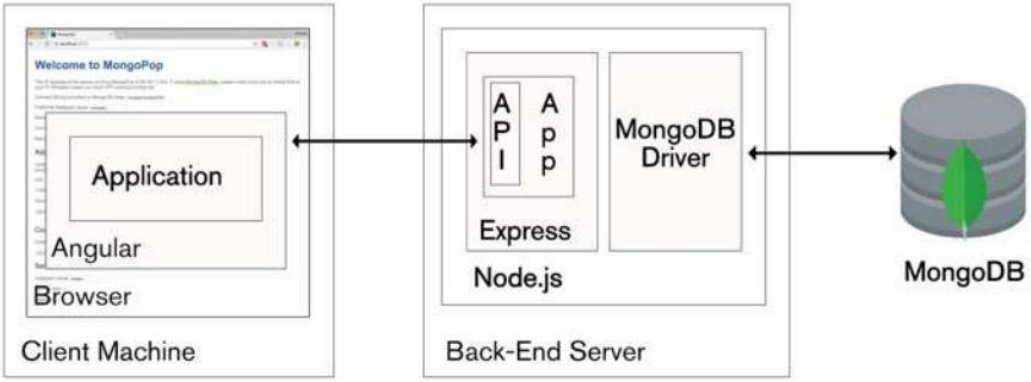
Arquitectura de MEAN

La arquitectura MEAN la cual se puede observar en la Figura 5 se compone de cuatro tecnologías JavaScript independientes. Está construida de tal manera que cada componente cumple una función específica mientras trabaja en conjunto con las otras capas para lograr un objetivo en común. En la arquitectura MEAN,

el cliente inicia las solicitudes, que fluyen desde la primera capa hasta la última. Las respuestas apropiadas a esas solicitudes luego fluyen de regreso al cliente a través de todos los niveles (Sharma et al.,2022).

Figura 5

Arquitectura MEAN



Nota. Tomado de (Sharma, Bajpai, Maheshwari, Sharma, & Gupta, 2022)

La interfaz de usuario, diseñada con el framework frontend Angular, es el primer punto de contacto con el cliente. El motor del lado del servidor, Node.js, recibe las solicitudes enviadas a través de la interfaz. Después, el framework Express.js envía una solicitud a la base de datos MongoDB. Express.js obtiene la respuesta de la base de datos en forma de datos. Finalmente, Node.js devuelve esta respuesta al cliente a través de la pantalla del usuario. (Sharma et al., 2022).

Selección técnica del stack tecnológico

Con base en los conceptos anteriormente detallados, se presenta a continuación una comparativa técnica en la Tabla 8 y un análisis de la selección del stack MEAN como entorno de desarrollo tecnológico para el sistema de gestión de turnos de la Cooperativa de Ahorro y Crédito Tulcán Ltda., considerando los requerimientos funcionales, las necesidades técnicas del proyecto y las características fundamentales de otros stacks comúnmente empleados en el desarrollo de aplicaciones web.

Puesto que el proyecto requiere una arquitectura flexible, entregas rápidas, capacidad de escalabilidad, alta interactividad con el usuario final y eficiencia en el manejo de datos, el stack MEAN se posiciona como la alternativa tecnológica

más adecuada frente a otras opciones como LAMP o Django. Estudios recientes señalan que el uso de MEAN facilita la integración completa entre frontend y backend en un solo lenguaje, mejora el rendimiento en entornos dinámicos y soporta el desarrollo de aplicaciones de una sola página (SPA) con respuestas rápidas y escalables (Sharma et al., 2022; Pratyusha et al., 2023; Khan et al., 2023).

Se concluye que la adopción del stack MEAN contribuye a alcanzar un desarrollo más eficiente y mantenible, alineado con las necesidades operativas y de experiencia del usuario propias del proyecto.

Tabla 8

Comparativa técnica entre stacks tecnológicos

Criterio técnico	**Stack LAMP (PHP, MySQL)**	**Django (Python, PostgreSQL)**	**Stack MEAN (MongoDB, Express, Angular, Node.js)**
Lenguaje principal	Múltiples (PHP, SQL, HTML)	Python y SQL	JavaScript (frontend, backend y base de datos)
Modelo de base de datos	Relacional (MySQL)	Relacional (PostgreSQL)	NoSQL (MongoDB – flexible y escalable)
Curva de aprendizaje	Alta (tecnologías heterogéneas)	Moderada	Moderada (todo en JavaScript)
Rendimiento y escalabilidad	Medio, adecuado para proyectos pequeños	Alto, adaptable a soluciones estructuradas	Alto, eficiente para proyectos modulares y escalables
Interactividad del usuario final	Limitada, páginas estáticas	Moderada, plantillas renderizadas en servidor	Alta, SPA con Angular y datos en tiempo real
Tiempos de desarrollo	Largos, por separación de tecnologías	Intermedios	Cortos, reutilización del lenguaje en todo el stack
Facilidad de mantenimiento	Baja, cambios afectan múltiples capas	Alta, gracias a la estructura de Django	Alta, modularidad y entorno homogéneo en JavaScript
Soporte de comunidad y recursos	Amplio, tecnologías maduras	Amplio, especialmente en proyectos académicos	Muy amplio y en crecimiento constante

Criterio técnico	Stack LAMP (PHP, MySQL)	Django (Python, PostgreSQL)	Stack MEAN (MongoDB, Express, Angular, Node.js)
Adaptabilidad al cambio	Limitada	Estructurada, menos flexible	Alta, ideal para ciclos iterativos y mejoras continuas
Reutilización y portabilidad	Baja, cada capa requiere perfiles distintos	Media	Alta, componentes reutilizables en cliente y servidor

Nota. Adaptado de Sharma et al. (2022); Pratyusha et al. (2023); Khan et al. (2023).

API REST

Una Interfaz de Programación de Aplicaciones (API) consiste en un conjunto de reglas y especificaciones que rigen la forma en que las aplicaciones de software se comunican entre sí. Este conjunto de reglas actúa como un contrato entre el consumidor o usuario y el proveedor, definiendo la estructura de las peticiones que el usuario puede realizar y el formato en que el proveedor debe entregar las respuestas a dichas peticiones (Castillo et al.,2022).

La arquitectura REST (Representational State Transfer) surgió como una nueva forma de concebir la interacción con Servicios Web. Los servicios basados en REST buscan emular el funcionamiento del protocolo HTTP, utilizando un grupo limitado de rutinas estándar como GET, POST, DELETE y PUT para gestionar los recursos que poseen un estado. Este enfoque ha transformado profundamente el panorama del desarrollo de software en los últimos años, facilitando la creación de aplicaciones web más escalables, flexibles y fáciles de integrar (Alamilla et al.,2021).

Una API RESTful es una interfaz que sigue la arquitectura REST para responder a solicitudes del cliente. Cuando un cliente solicita a una API RESTful, ésta le envía el estado del recurso. La respuesta se envía por HTTP en formatos como JSON, HTML, XML o texto plano (Castillo et al.,2022).

Debido a su simplicidad, escalabilidad y flexibilidad, las APIs RESTful se han convertido en un estándar en el desarrollo de aplicaciones web, permitiendo el desarrollo de aplicaciones que se integran con facilidad a otros sistemas y plataformas.

2.3. MARCO LEGAL

El marco legal define las reglas y regulaciones que regulan el uso de tecnologías en la administración de servicios en entidades financieras en Ecuador. En este marco, la aplicación del stack MEAN en un sistema de gestión de turnos debe respetar normativas legales que aseguren la calidad del servicio, protección de datos y eficacia en la atención al cliente. Se enumeran las normativas principales que son aplicables:

2.3.1. Constitución de la República del Ecuador

La Constitución de la República del Ecuador (2008) como norma jurídica fundamental del país, establece en su artículo 52 que: “Las personas tienen derecho a disponer de bienes y servicios de óptima calidad y a elegirlos con libertad, así como a una información precisa y no engañosa sobre su contenido y características.” Así como también, el número 25 del artículo 66 reconoce y asegura el derecho de las personas a acceder a bienes y servicios públicos y privados de calidad, con eficiencia, eficacia y buen trato, además de recibir información real y adecuada sobre su contenido y características.

2.3.2. Ley Orgánica de Defensa del Consumidor

El número 2 del artículo 4 de la Ley Orgánica de Defensa del Consumidor (2000) indica que son derechos del consumidor a más de los establecidos en la Constitución Política de la República, el derecho a que proveedores públicos y privados oferten bienes y servicios competitivos, de óptima calidad, y a elegirlos con libertad.

2.3.3. Ley de Comercio Electrónico, Firmas Electrónicas y Mensajes de Datos

Dado que los sistemas de gestión de turnos pueden operar a través de medios digitales, es relevante considerar la Ley de Comercio Electrónico, Firmas Electrónicas y Mensajes de Datos. Registro Oficial No. 557 (2002). Esta normativa regula la validez jurídica de la información electrónica y los servicios digitales. En particular, el artículo 5 establece que los datos electrónicos tienen

la misma validez legal que los documentos físicos, lo cual es crucial para la digitalización de los procesos en instituciones financieras.

2.3.4. Código Orgánico Monetario y Financiero

El Código Orgánico Monetario y Financiero (2022) en la sección 4, artículo 153 menciona que: “La Junta de Política y Regulación Monetaria y Financiera regulará los estándares de calidad de los servicios financieros, de los sistemas de medición de satisfacción de los clientes y usuarios y de los sistemas de atención y reparación. Las entidades financieras prestarán servicios eficientes, oportunos y transparentes, para lo cual informarán a los usuarios y/o clientes, a través de los diferentes canales de comunicación que mantengan, sobre los servicios y cargos, de acuerdo con las normas y frecuencia establecidos para el efecto.”

CAPÍTULO III

METODOLOGÍA

3.1. DESCRIPCIÓN DEL ÁREA DE ESTUDIO/GRUPO DE ESTUDIO

El área de estudio de esta investigación se desarrolla en la Cooperativa de Ahorro y Crédito Tulcán Ltda., una entidad financiera fundada en 1968 tiene su matriz en Tulcán, provincia del Carchi, Ecuador, además de poseer una amplia red de agencias en la sierra ecuatoriana que abarca ciudades como Julio Andrade, Huaca, San Gabriel, El Ángel, Ibarra, Otavalo, Quito, Latacunga, Guaranda y Ambato, estas ubicaciones geográficas dentro del territorio ecuatoriano pueden ser observadas en la Tabla 9. Su objetivo es brindar servicios financieros accesibles y de calidad a sus socios y clientes, impulsando el desarrollo económico y social de la región. (Cooperativa de Ahorro y Crédito Tulcán Ltda., 2023, 3 de marzo).

La gama de productos financieros ofrecidos por la cooperativa incluye cuentas de ahorro, préstamos personales, depósitos a plazo fijo, créditos para vivienda y microcréditos para emprendedores. El grupo de estudio está conformado por los socios y colaboradores de la Cooperativa de Ahorro y Crédito Tulcán, quienes provienen de diversos sectores socioeconómicos y tienen distintos objetivos financieros. La composición de los colaboradores abarca diferentes edades y niveles educativos, al provenir de diferentes partes y sectores del país, reflejando la diversidad de perfiles dentro de la comunidad financiera de la institución.

El personal encargado de la atención al cliente dentro de la Cooperativa de Ahorro y Crédito Tulcán que incluye personal de cajas y servicios cooperativos quienes controlan y administran el servicio brindado por la institución en cada una de sus agencias serán quienes brinden la información y datos a considerar en la presente investigación.

Esta selección se fundamenta tomando en cuenta al personal que interactúa diariamente con los clientes de la institución brindándoles información o ayuda con cada uno de sus requerimientos de los canales electrónicos y servicios presenciales que se brindan dentro de las agencias de la institución.

Tabla 9

Agencias Cooperativa de Ahorro y Crédito Tulcán

Agencia	Dirección	Provincia
Oficina Matriz	Sucre y Atahualpa esquina	Carchi
Agencia Tulcán	Sucre y Atahualpa esquina	Carchi
Agencia Julio Andrade	Pichincha entre García Moreno y Juan Montalvo	Carchi
Agencia Huaca	8 de Diciembre y Juan Montalvo	Carchi
Agencia San Gabriel	Sucre y Bolívar (parque principal)	Carchi
Agencia El Ángel	Esmeraldas y Pichincha (junto a EMAPA-E)	Carchi
Agencia Ibarra	Sucre y Oviedo esquina	Imbabura
Agencia Quito Centro	Av. 10 de Agosto 2380 y Alonso de Mercadillo	Pichincha
Agencia Quito Sur	Av. Pedro Vicente Maldonado y Guayanay Ñan (Quitumbe)	Pichincha
Agencia Quito Norte	Av. La Prensa, CC Las Violetas Loc. 1-2 (El Condado)	Pichincha
Agencia Latacunga	Calle Quijano y Ordoñez y Félix Valencia (La Merced)	Cotopaxi
Agencia Sangolquí	Av. Luis Cordero y Bolívar (frente al River Mall)	Pichincha
Agencia Cayambe	Calle Terán y Junín (a 1 cuadra del Parque Central)	Pichincha
Agencia Otavalo	Morales entre Bolívar y Sucre	Imbabura
Agencia Ambato	Av. Cevallos y Unidad Nacional esquina	Tungurahua
Agencia Riobamba	Calle Veloz entre Colón y Espejo	Chimborazo
Agencia Guaranda	Calle Olmedo entre Convención de 1884 y Sucre	Bolívar

Nota. Tomado de Cooperativa de Ahorro y Crédito Tulcán Ltda. (2024), Red de agencias. https://www.cooptulcan.fin.ec/red-de-agencias/

3.1.1. Población y Muestra

La investigación abarca a los usuarios y al personal de la Cooperativa de Ahorro y Crédito Tulcán Ltda., que aplicará el sistema de gestión de turnos. Los datos sobre usuarios atendidos diariamente y el personal que gestiona turnos se obtuvieron de registros internos de la cooperativa y entrevistas con los encargados de atención al cliente.

Se aplicó la técnica de muestreo no probabilístico intencional para elegir participantes que cumplan criterios específicos para la investigación. De acuerdo con Otzen y Manterola (2017), esta técnica es apropiada para poblaciones heterogéneas que necesitan un enfoque específico para un grupo con características particulares. La muestra incluyó empleados que manejan turnos y clientes que usaron el sistema, facilitando la evaluación de su eficiencia y satisfacción. La recolección de información y el análisis se desarrollan durante el año 2024.

3.2. ENFOQUE Y TIPO DE INVESTIGACIÓN

3.2.1. Enfoque

El enfoque mixto combina la metodología cuantitativa y la metodología cualitativa, basándose en el propósito del estudio y la naturaleza de la pregunta de investigación, permite así, un entendimiento más profundo y completo del fenómeno a estudiar. Según Taherdoost (2022), además, el enfoque mixto puede centrarse en ambos métodos de manera equitativa o priorizar uno sobre el otro, dependiendo del proceso de integración seleccionado.

El empleo de un enfoque mixto en esta investigación se justifica por la naturaleza dual del objeto de estudio, el sistema basado en el stack MEAN requiere tanto de análisis cuantitativo, medición objetiva de tiempos de operación y eficiencia del sistema, así como también de evaluaciones cualitativas que abarcan percepciones y opiniones de los usuarios y del personal de la institución. Esta complementariedad metodológica permite, validar estadísticamente las mejoras operativas y comprender los factores humanos que determinan la adopción exitosa de la tecnología, proporcionando así una evaluación integral que responde tanto a criterios técnicos como organizacionales en el contexto de las cooperativas de ahorro y crédito.

Este método proporciona una comprensión más amplia del fenómeno analizado, al aprovechar la profundidad de la información cualitativa y la objetividad de la información cuantitativa. Además, el enfoque mixto fomenta la triangulación de los resultados, lo que mejora la validez y la confiabilidad de las conclusiones alcanzadas, al ofrecer múltiples perspectivas sobre el mismo problema de investigación.

3.2.2. Tipo de investigación

Este trabajo se fundamenta en una investigación descriptiva, utilizada para recolectar datos, describirlos y analizarlos posteriormente. La investigación descriptiva busca detallar individuos, eventos o condiciones tal como aparecen en la naturaleza, sin alterar las variables. Según Siedlecki (2020), este tipo de investigación es útil para explorar las características de una población, identificar problemas existentes o analizar variaciones en prácticas entre instituciones o grupos. Este enfoque es adecuado para este estudio porque permite recopilar datos sobre el desempeño de la implementación de un marco tecnológico con el uso del stack MEAN aplicado a la gestión de turnos en cooperativas de ahorro y crédito. Asimismo, proporciona datos útiles sobre las expectativas de clientes y trabajadores, información que puede ser fundamental para futuras investigaciones.

Las fuentes para recopilar datos incluirán observación, encuestas y análisis documental. Estas herramientas permiten evaluar el impacto de la solución tecnológica propuesta, centrada en el stack MEAN (MongoDB, Express.js, Angular y Node.js), en la eficiencia operativa. La investigación descriptiva es valiosa aquí, ya que describe la gestión de turnos en la cooperativa e identifica áreas de mejora y oportunidades para optimizar procesos.

Además del enfoque descriptivo, este estudio hace uso de una investigación exploratoria, que se utiliza para explorar nuevas oportunidades y desafíos que surgen durante la implementación del software de gestión de turnos basado en el stack MEAN en la Cooperativa de Ahorro y Crédito Tulcán. Esta investigación exploratoria se llevó a cabo mediante la revisión de la literatura especializada en tecnología y gestión de servicios.

La combinación de ambos métodos de investigación ofrece una visión más abarcadora y profunda del efecto de la tecnología en las cooperativas de ahorro y crédito y en sus clientes.

3.3. DEFINICIÓN Y OPERACIONALIZACIÓN DE VARIABLES

Existe una relación significativa entre la implementación del marco tecnológico con el uso del stack MEAN en la administración de turnos en cooperativas de ahorro y crédito y la eficiencia operativa, así como la satisfacción del cliente. Para analizar esta relación, en la Tabla 10 se presentan las variables principales del estudio, su clasificación y la forma en que serán medidas.

La variable independiente, Implementación del marco tecnológico con el uso del stack MEAN, se refiere al desarrollo del sistema utilizando MongoDB, Express.js, Angular y Node.js para optimizar la gestión de turnos en cooperativas. Su valoración se basa en observar el proceso de desarrollo, la implementación técnica y la revisión de documentación.

Por otro lado, la variable dependiente, mide la reducción del tiempo de espera promedio, el porcentaje de turnos atendidos puntualmente y la percepción del usuario. Para ello, se emplean técnicas de observación directa y encuestas, complementadas con el análisis de sugerencias y quejas registradas, así como la facilidad de uso del sistema.

La implementación de estas variables organiza el análisis de la investigación, garantizando que los datos obtenidos reflejen objetivamente el efecto del stack MEAN en la gestión de turnos en la cooperativa de ahorro y crédito.

Tabla 10

Definición de variables

Variable	Tipo y Dimensión de Variable	Indicadores	Técnica	Instrumento
Implementación del marco tecnológico con el uso del stack MEAN	Independiente (Cualitativa Nominal) **Desarrollo del sistema**	Uso de MongoDB para base de datos. Uso de Express.js en la gestión del backend. Desarrollo del frontend con Angular. Uso de Node.js como entorno de ejecución. Tiempo de desarrollo del sistema. Nivel de integración con otros sistemas.	Implementación técnica. Observación del proceso de desarrollo, revisión de documentación técnica.	Observación del proceso de desarrollo del sistema, revisión de documentación técnica.
Eficiencia operativa	Dependiente (Cuantitativa Discreta) **Eficiencia en la gestión de turnos**	Tiempo de espera promedio (medido en minutos). Porcentaje de turnos atendidos puntualmente (según un umbral de tiempo establecido).	Observación directa	Registro de tiempos de espera, seguimiento de la puntualidad en la atención.

3.4. PROCEDIMIENTOS

Para realizar la metodología de investigación sobre la implementación del marco tecnológico basado en el stack MEAN para gestionar turnos en cooperativas de ahorro y crédito, se ha creado un proceso dividido en tres fases claramente definidas. Cada una de estas etapas se sincroniza con un objetivo específico, lo que posibilita un examen exhaustivo y una evaluación rigurosa de los datos y modelos empleados en el estudio. Cada fase incluye procedimientos específicos de recolección, análisis, diseño y validación que aseguran un enfoque sistemático y replicable.

Fase 1. Análisis del proceso, métodos y tecnologías de gestión de turnos en cooperativas de ahorro y crédito ecuatorianas

El estudio en su fase inicial se centró en describir los procesos actuales de gestión de turnos en diez cooperativas de ahorro y crédito, abarcando un análisis pormenorizado de la Cooperativa de Ahorro y Crédito Tulcán Ltda. Como parte del procedimiento, se seleccionaron cooperativas con base en su tamaño, ubicación y nivel tecnológico, y se diseñaron instrumentos como entrevistas semiestructuradas y encuestas estructuradas. Se aplica una metodología que recolecta datos primarios a través de entrevistas al personal operativo y encuestas a usuarios, para documentar métodos, tecnologías en uso y desafíos principales. Paralelamente, se evalúa el impacto operativo de estos sistemas en indicadores clave como tiempos de espera y eficiencia en la atención. La información recolectada fue procesada mediante herramientas como Excel, aplicando técnicas estadísticas descriptivas y correlacionales. Además, se generaron gráficos de visualización que ayudaron a detectar patrones. Estos procedimientos permiten establecer una base comparativa que guía el diseño posterior de la solución tecnológica. Los resultados logrados ofrecen una base firme para el diseño de un marco tecnológico ajustado a las necesidades del sector cooperativo ecuatoriano.

Fase 2. Diseño y desarrollo de un sistema con el uso del stack MEAN para la gestión de turnos en la Cooperativa de Ahorro y Crédito Tulcán.

La segunda etapa se centra en la creación y diseño de un sistema con el stack MEAN. Comienza con la definición de los requisitos del sistema, fundamentados en la petición de los líderes de la Cooperativa de Ahorro y Crédito Tulcán Ltda., donde se ejecutará el sistema. Para ello, se llevaron a cabo reuniones técnicas con usuarios clave, donde se definieron los casos de uso, flujos operativos y requerimientos técnicos. Posteriormente, se lleva a cabo la implementación del sistema con las tecnologías del stack MEAN: MongoDB para la base de datos, Express.js para el backend, Angular para el frontend, Node.js para el entorno de ejecución y se incorpora socket.io para la comunicación en tiempo real entre dispositivos. Durante la implementación, se aplicaron procedimientos de desarrollo iterativo siguiendo la metodología Scrum, estructurando el trabajo en sprints de dos semanas. Durante esta fase, se realiza un proceso iterativo de revisión con el usuario y desarrollo basado en la metodología SCRUM, asegurando la alineación del sistema con las necesidades y requisitos de la Cooperativa de Ahorro y Crédito Tulcán. Además, se ejecutaron pruebas unitarias, de integración y de aceptación por parte de usuarios, como parte de los procedimientos de validación funcional del sistema.

Fase 3. Propuesta del marco tecnológico basado en el stack MEAN.

Esta fase final del estudio tiene como objetivo principal formular un marco tecnológico integral para la gestión de turnos, fundamentado en el stack MEAN (MongoDB, Express.js, Angular y Node.js) complementado con Socket.io para comunicación en tiempo real. Se sistematizaron los aprendizajes técnicos del desarrollo y se estructuró una propuesta compuesta por cuatro pilares: arquitectura, modelo de datos, flujos operativos y mecanismos de integración. La propuesta se estructura en cuatro componentes esenciales: arquitectura técnica, modelo de datos, flujos operativos y mecanismos de integración. El marco propuesto incorpora un diseño modular que permite su adaptación a diferentes contextos organizacionales del sector cooperativo, garantizando escalabilidad y mantenibilidad. Como procedimiento adicional, se elaboró una guía técnica que documenta los pasos de implementación del marco propuesto, incluyendo

estándares de desarrollo, protocolos de comunicación y criterios de escalabilidad. Se especifican los protocolos de comunicación entre componentes y los estándares de desarrollo, estableciendo así un referente completo para la modernización de sistemas de gestión de turnos en cooperativas de ahorro y crédito.

3.4.1. Técnicas de investigación

Este estudio emplea varias técnicas de investigación: se realizó una encuesta a los empleados operativos y técnicos de cooperativas de ahorro y crédito, para entender el estado actual de la gestión de turnos en este sector, sus limitaciones y áreas de mejora. Este enfoque ofrece una perspectiva completa de los retos y oportunidades que tienen estas instituciones, permitiendo un análisis más exhaustivo de su proceso operativo y administrativo.

Asimismo, se llevó a cabo una observación directa en la Cooperativa de Ahorro y Crédito Tulcán Ltda., lo que facilitó el análisis del proceso de gestión de turnos desde la perspectiva de los clientes y el funcionamiento en el área técnica. Asimismo, se obtuvo información del flujo de atención y manejo de turnos administrativos, proporcionando una visión amplia de la situación actual y la posibilidad de aplicar este marco tecnológico en el futuro.

Como resultado de estas observaciones, se identificaron problemas específicos como tiempos de espera prolongados en horas de mayor afluencia, duplicación de turnos en distintos módulos, ausencia de mecanismos de retroalimentación directa del cliente y carencia de información clara en pantalla sobre el orden de atención. Los usuarios manifestaron insatisfacción por estas deficiencias. Estos hallazgos se utilizaron como base para definir funcionalidades clave del sistema propuesto, como la visualización en tiempo real del estado de atención, notificaciones del siguiente turno y registro de la experiencia del cliente.

3.5. CONSIDERACIONES BIOÉTICAS

Este estudio se lleva a cabo bajo estrictos principios éticos, asegurando la protección de la información y la privacidad de los participantes, lo cual es crucial en la recopilación de datos sensibles. Debido a que la investigación se sustenta en encuestas y datos de clientes, es crucial obtener el consentimiento informado

de los participantes, quienes deben entender completamente el objetivo del estudio antes de ofrecer cualquier dato, asegurando su participación voluntaria. Para salvaguardar la confidencialidad y la privacidad, los datos personales que se recopilan serán anonimizados y utilizados únicamente para fines estadísticos y de mejora del sistema, evitando la identificación de los individuos involucrados en el estudio. Además, se implementará un uso responsable de la información recopilada, que será protegida bajo estándares de seguridad apropiados; no se divulgará a terceros sin el consentimiento de los participantes. Así, se pretende adherir a los principios de beneficencia y transparencia, centrando en optimizar la eficiencia en la gestión de turnos y la experiencia del usuario, sin afectar la integridad de los participantes.

CAPÍTULO IV

RESULTADOS Y DISCUSIÓN

4.1. ENCUESTA DE DIAGNÓSTICO

Se desarrolló una encuesta que se enfocó en el estado actual de la tecnología en la gestión de turnos dentro de diversas cooperativas de ahorro y crédito en Ecuador. El objetivo principal fue determinar si estas entidades cuentan con sistemas automatizados para este proceso y conocer los desafíos más comunes que enfrentan. Asimismo, se buscó comprender cómo se gestiona actualmente la asignación de turnos en el sector cooperativo del país.

La encuesta fue aplicada a personal técnico de cooperativas de ahorro y crédito evidencia dos problemas críticos como se puede observar en la Tabla 11 existe una falta de integración de sistemas mencionado en el 100% de los casos y gestión manual ineficiente asociada a tiempos de espera prolongados. Estos hallazgos destacan la necesidad urgente de soluciones tecnológicas que prioricen interoperabilidad y automatización.

Tabla 11

Entidades Financieras encuestadas

Nombre de la Entidad Financiera	Tipo de gestión de turnos	Principales desafíos	Características esenciales requeridas
Cooperativa de Ahorro y Crédito Pablo Muñoz Vega	Sistema automatizado	Falta de integración	Integración con otros sistemas, Notificaciones, Reportes
Cooperativa de Ahorro y Crédito Educadores Tulcán Ltda.	Manual	Tiempos de espera prolongados	Notificaciones en tiempo real

Nombre de la Entidad Financiera	Tipo de gestión de turnos	Principales desafíos	Características esenciales requeridas
Cooperativa de Ahorro y Crédito Cooprogreso	Sistema automatizado	Saturación de turnos	Integración con otros sistemas, Reportes
Cooperativa de Ahorro y Crédito Coopad	Manual	Tiempos de espera prolongados	Notificaciones en tiempo real
Cooperativa de Ahorro y Crédito Luz del Valle	Manual	Falta de integración	Integración, Notificaciones, Reportes
Cooperativa de Ahorro y Crédito Tulcán	Sistema automatizado	Tiempos de espera prolongados	Integración con otros sistemas, Reportes
Cooperativa de Ahorro y Crédito 23 de Julio	Manual	No tienen software	Integración, Notificaciones, Reportes
Cooperativa de Ahorro y Crédito 29 de octubre	Mixto	Falta de integración	Integración con otros sistemas
Cooperativa de Ahorro y Crédito Artesanos	Manual	Falta de integración y control	Integración, Notificaciones, Reportes

Las figuras siguientes muestran los resultados y análisis de las diversas preguntas tratadas en la encuesta junto a su análisis respectivo. A través de estas representaciones gráficas, se ilustran de manera clara y precisa los hallazgos obtenidos, permitiendo una comprensión visual de los datos.

4.1.2. Preguntas de la encuesta

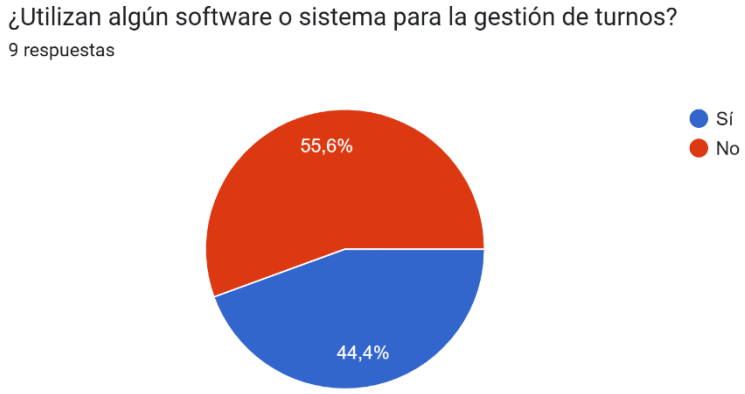
Figura 6

Pregunta 1 – Encuesta

En los resultados se puede observar que más de la mitad de las instituciones realizan manualmente los procesos de gestión de turnos, lo que causa problemas operativos y se traduce en tiempos y costos de gestión mucho más altos.

Figura 7

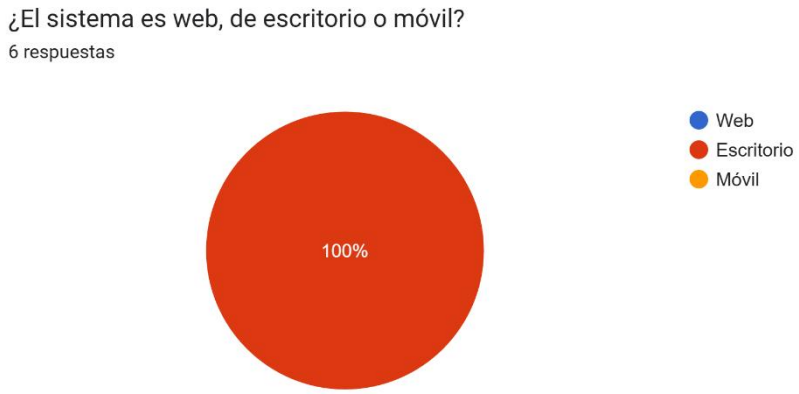
Pregunta 2 – Encuesta



Las cifras indican que la mayoría de las instituciones no hace uso de software para la gestión de turnos, mientras que una pequeña parte de ellas ha adoptado algún tipo de software automatizado. Por lo tanto, nace la posibilidad de adaptar este proceso a tecnología de vanguardia e implementar software moderno tipo MEAN Stack.

Figura 8

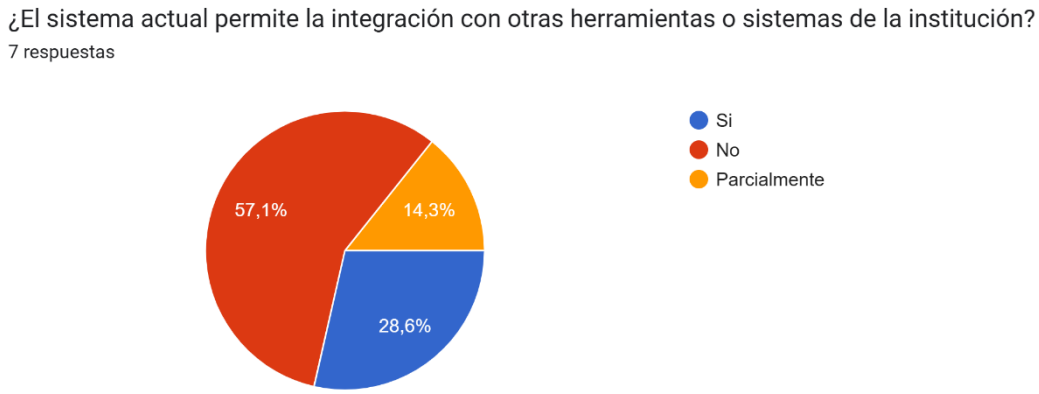
Pregunta 3 – Encuesta



El resultado indica que los sistemas utilizados por las cooperativas analizadas son de escritorio, lo cual limita el acceso por parte de los clientes al no contar con plataformas web o móviles. Desarrollar la solución utilizando el stack MEAN, al ser una tecnología orientada a la web, permitiría mayor flexibilidad y accesibilidad.

Figura 9

Pregunta 4 – Encuesta

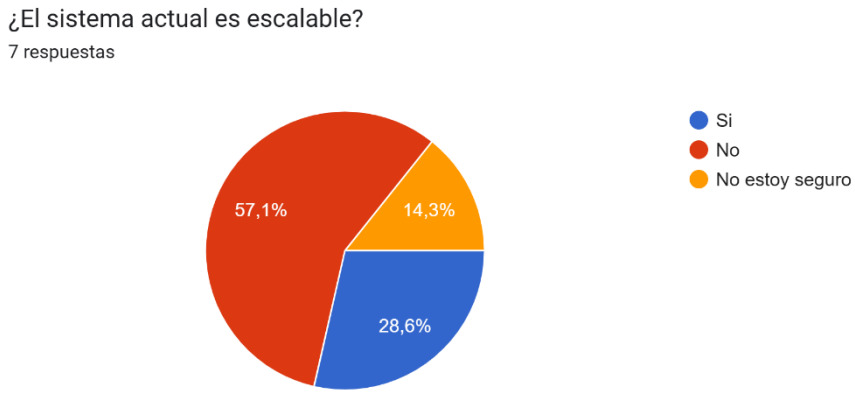


Solamente una pequeña porción de las instituciones responde que sus sistemas permiten conectividad con otras herramientas, mientras que la mayoría no lo hace. Esto hace que los procesos operativos no sean nada fluidos. En el caso

de Stack MEAN, consiguiendo la máxima integración, se pueden relacionar sistemas y procesos.

Figura 10

Pregunta 5 – Encuesta



Solo una pequeña parte de las instituciones considera que su sistema es escalable y el resto de ellas no está seguro o no lo es. Esta falta de escalabilidad limita el crecimiento futuro. El Stack MEAN, al ser modular y escalable, permite adaptarse a las necesidades cambiantes de las instituciones.

Figura 11

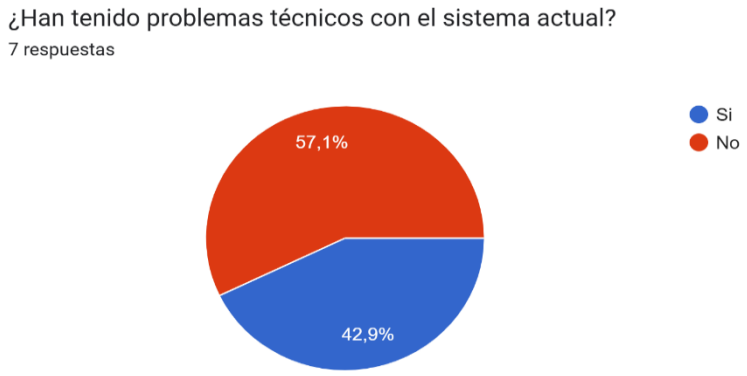
Pregunta 6 – Encuesta

Los resultados evidencian que son pocas las instituciones que tienen una base de datos integrada, esta falta dificulta el análisis de datos y la toma de decisiones.

Stack MEAN con MongoDB como base de datos, permitiría consolidar la información en una sola plataforma.

Figura 12

Pregunta 7 – Encuesta



Una gran parte de las instituciones reporta problemas técnicos repetitivos, especialmente aquellas con sistemas automatizados. Estos problemas incluyen fallas de integración y falta de estabilidad.

Figura 13

Pregunta 8 – Encuesta

En la mayoría de las instituciones se observa que los sistemas utilizados son provistos por proveedores externos, lo que genera una dependencia hacia terceros. El uso del stack MEAN, al ser de código abierto y emplear un mismo lenguaje en todo su ecosistema, permitiría a las instituciones desarrollar y

mantener sus propias soluciones, reduciendo costos y aumentando su autonomía.

Figura 14

Pregunta 9 – Encuesta

El análisis de los desafíos considerados en el estudio indica que en todas las cooperativas es preciso mejorar la gestión de turnos. La ruptura de la integridad hace la coordinación entre áreas difícil, los tiempos largos son el resultado de una mala gestión del tiempo del usuario. Un exceso de turnos indica una mala gestión de la demanda.

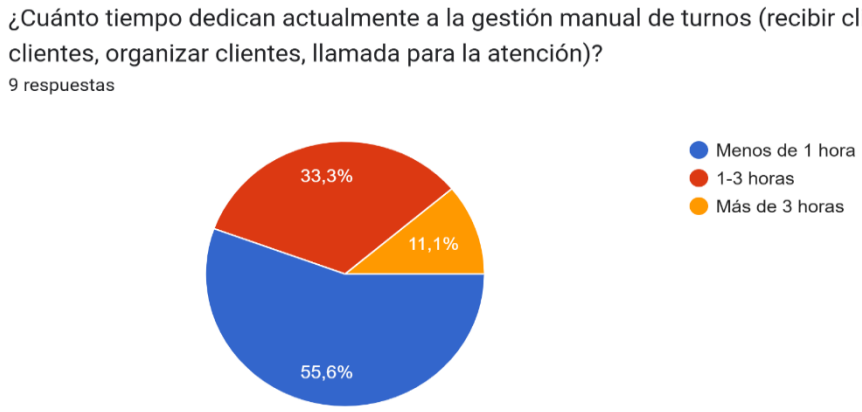
Figura 15

Pregunta 10 – Encuesta

Según todas las entidades, un sistema más innovador les podría aportar un mejor funcionamiento en la gestión de turnos. Este ofrecimiento hacia la innovación es suficiente, y a la vez va en la línea de una solución que funcione sobre el Stack MEAN, que promete modernidad, eficiencia y escalabilidad.

Figura 16

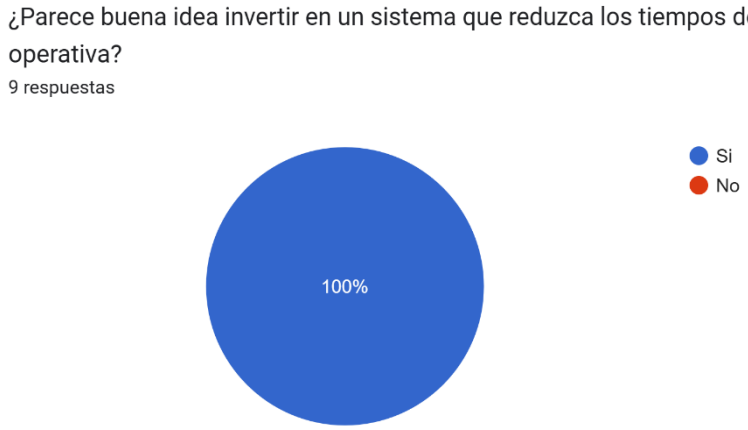
Pregunta 11 – Encuesta



El diagnóstico de la carga operativa revela que un porcentaje elevado de las organizaciones, todavía hoy, dedican un tiempo elevado a la modalidad manual de la gestión de los turnos, lo que tiene su trascendencia en lo que concierne a la eficiencia y la utilización de recursos. La heterogeneidad en el tiempo que al proceso le dedican sugiere diferencias en los procesos internos y también en el nivel de digitalización que hay en cada entidad. La implementación de un sistema automatizado mejoraría dicha situación, en la medida en que reduciría el esfuerzo administrativo, trasladando el foco hacia un mayor trabajo en la atención al cliente.

Figura 17

Pregunta 12 – Encuesta



La aceptación unánime que ha tenido la cuestión de la inversión en un modernizado sistema exprime a las claras una notoria necesidad de modernizar la gestión de los turnos. Esta disposición da a entender que los centros son conscientes de las insuficiencias de sus sistemas actuales y están abiertos a soluciones tecnológicas.

4.1.3. Análisis e Interpretación de resultados

Gestión de Turnos Mayormente Manual: La mayoría de las cooperativas de ahorro y crédito encuestadas gestionan los turnos de manera manual, lo que implica un alto grado de intervención humana en procesos como la recepción de clientes, organización de turnos y redireccionamiento. Este enfoque manual no solo consume tiempo entre 1 y 3 horas diarias en algunos casos según los datos, sino que también tiende a ser propenso a errores humanos especialmente en horas pico al ser manejado especialmente de forma operativa.

Falta de Sistemas Automatizados: Sólo un pequeño porcentaje de las instituciones utiliza sistemas automatizados para la gestión de turnos, y en su mayoría, estos sistemas son de escritorio y no están integrados con otras herramientas y tampoco manejan una base de datos única entre todas sus agencias o sucursales. Esto limita la capacidad de las instituciones para realizar análisis de datos consolidados y decidir basándose en datos en tiempo real.

Desafíos Principales: Una vez finalizada la fase de evaluación se definieron los principales retos que enfrentan las instituciones haciendo alusión a la saturación de turnos, tiempos de espera excesivos y falta de integración entre sistemas. Problemas que impactan directamente la operabilidad y satisfacción del cliente, dado que tiempos de espera demasiado altos pueden ocasionar descontento y pérdida de clientes.

Necesidad de Modernización: La mayoría de las instituciones económicas que manejan un sistema para la gestión de turnos recurren a uno de escritorio, con las desventajas que conlleva frente a un sistema web. La mayoría de las instituciones hacen notar la importancia de un sistema más moderno y eficiente que a la postre mejoraría la gestión de turnos. Las características más admiradas son: integración con sistemas, notificaciones en tiempo real y generación de reportes automáticos. Las características provenientes de la integración entre sistemas proporcionarían una gestión más ágil de los turnos, pero también ofrecerían una mejora de la experiencia de los clientes.

Apertura a la Inversión en Tecnología: Las instituciones económicas encuestadas muestran una disposición favorable hacia la inversión en los sistemas que mejoren la gestión de turnos; se denota que existe un mercado receptivo para las soluciones tecnológicas que puedan abordar los problemas mencionados.

Los datos proporcionados sustentan la necesidad de llevar adelante la solución tecnológica adecuada a los problemas mencionados, o bien un sistema web basado en un stack MEAN. La implementación de este sistema además de aumentar la eficiencia de las operaciones de la organización impactaría positivamente en la experiencia del cliente y en la capacidad de las instituciones para decidir utilizando información cuantitativa. La mejora en la gestión de turnos es un aspecto crucial para las cooperativas de ahorro y crédito que buscan ser competitivas en un entorno financiero cambiante.

4.2. DESARROLLO DEL PROYECTO DE SOFTWARE

4.2.1. Estructura y Asignación de Roles en el Proyecto

Este proyecto se desarrolla de forma colaborativa en el que se cuentan con tres desarrolladores de la institución, en el cual el autor asume las funciones y responsabilidades propias del proceso, como planificar, diseñar, implementar y desarrollar en colaboración el sistema, asumiendo estas funciones para asegurar el buen funcionamiento del stack MEAN. Para ello, el proyecto también cuenta con un tutor que supervisa el avance, brinda una retrospectiva y verifica que se cumplen los objetivos establecidos, aplicando métodos y buenas prácticas en el desarrollo del sistema.

4.2.2. Requerimientos del sistema

Los requisitos funcionales y no funcionales son el resultado de explorar las expectativas y necesidades de los usuarios de la Cooperativa de Ahorro y Crédito Tulcán Ltda., donde se implementará el sistema. En el proceso se realizaron reuniones con los usuarios relevantes, así como la observación de los procesos operativos vigentes. También se tuvieron en cuenta las necesidades técnicas necesarias para asegurar la correcta integración y funcionamiento del sistema en la infraestructura existente en la institución y asegurando la satisfacción de los estándares de rendimiento, seguridad y escalabilidad requeridos. Estos requerimientos funcionales y no funcionales son detallados en el ANEXO A basándose en el estándar IEEE830.

4.2.3. Diagramas de Casos de Uso

Los casos de uso descritos aquí se han creado según los requisitos obtenidos de los usuarios de la Cooperativa de Ahorro y Crédito Tulcán Ltda. Estos casos de uso explican las interacciones clave entre los usuarios y el sistema, tratando los procesos que serán esenciales para el buen funcionamiento del sistema en el entorno operativo de la cooperativa. Cada caso de uso refleja las necesidades específicas de los usuarios, asegurando que el sistema satisfaga las expectativas tanto del lado técnico como funcional. Cada uno de los casos de uso son diagramados y explicados en el ANEXO A basándose en el estándar IEEE830.

4.2.4. Planificación y desarrollo de Sprints

La presente planificación presente en la Tabla 12 tiene como objetivo organizar los 26 Requerimientos Funcionales (RF) y 17 Requerimientos No Funcionales (RNF) en 10 sprints, considerando su prioridad, dependencias técnicas y tiempo estimado. Se siguen los lineamientos de Scrum, asignando una duración fija de 14 días por sprint para garantizar una entrega incremental y controlada. Durante este proceso se realizaron pruebas funcionales con el usuario realizando actas de prueba visibles en el ANEXO F y G.

Criterios de Priorización

1. **Alta prioridad (≥100):** Requerimientos esenciales para el funcionamiento básico del sistema (Sprints 1-3).
2. **Media prioridad (99-50):** Funcionalidades importantes, pero no críticas (Sprints 4-7).
3. **Baja prioridad (49-25):** Mejoras y funcionalidades adicionales (Sprints 8-10).

Tabla 12

Planificación de Sprints

Sprint	Nombre	Prioridad (Rango)	Requerimientos	Puntos	Duración
1	Fundamentos del Sistema	**Alta (100)**	RF01, RF03, RNF04, RNF05, RNF07, RNF01	19	14 días
2	Gestión de Usuarios	**Alta (98)**	RF02, RF10, RF25, RNF06, RNF16	18	14 días
3	Turnos Básicos	**Alta (95)**	RF04, RF05, RF08,	16	14 días
4	Turnos en Línea	**Media (90)**	RF07, RF19, RF21, RNF10	15	14 días

Sprint	Nombre	Prioridad (Rango)	Requerimientos	Puntos	Duración
5	Gestión Operativa	**Media (85)**	RF06, RF16, RF22, RNF17	14	14 días
6	Administración Avanzada	**Media (80)**	RF09, RF11, RF26, RNF02	15	14 días
7	Reportes y Estadísticas	**Media (75)**	RF13, RF17, RF15, RNF15	15	14 días
8	Mejoras de Usuario	**Baja (60)**	RF12, RF14, RF18, RNF11	16	14 días
9	Seguridad y Validaciones	**Baja (55)**	RF20, RF23, RF24, RNF03, RNF12, RNF13	16	14 días
10	Optimizaciones Finales	**Baja (50)**	RNF08, RNF09, RNF14	12	14 días

Esta planificación garantiza una implementación estructurada del sistema, priorizando funcionalidades críticas en los primeros sprints y dejando mejoras y optimizaciones para etapas posteriores. Cada sprint está balanceado en carga de trabajo y asegura una entrega incremental y funcional del software.

A continuación, se presentan los desarrollos e implementaciones realizadas durante cada uno de los Sprint planificados.

Inicio Sprint 1: Fundamentos del Sistema

Para cada uno de los requerimientos se pueden revisar detalles en el respectivo identificador del requerimiento en el desarrollo del sistema del ANEXO C.

Creación y diseño de base de datos (RNF01)

Se desarrolla el diagrama de la base de datos el cual representa la estructura de datos del software de gestión de turnos. Esta se encuentra diseñada para manejar información relacionada con los turnos, usuarios, oficinas, servicios, módulos, horarios, y otros elementos necesarios para la operación del sistema, se utiliza la base de datos NoSQL (MongoDB), dentro del ANEXO C en el RNF01 se puede visualizar y obtener detalles adicionales sobre el diagrama, colecciones principales y sus respectivos atributos y relaciones.

Colecciones de la base de datos

El sistema se estructura en 11 colecciones clave que garantizan su completa operatividad:

- **AuditoriaTurno:** registra cambios de estado en los turnos que ingresan al sistema.
- **Parametro**: permite configuraciones dinámicas dentro del sistema, como por ejemplo cuántos intentos son permitidos en un OTP incorrecto.
- **Role:** permite implementar RBAC con perfiles como "administrador" o “servicios”.
- **TurnoEnLinea:** gestiona e informa de los turnos digitales que son tomados vía internet mediante la autenticación en dos pasos por OTP.
- **User:** centraliza la información primordial de cada uno de los usuarios.
- **Oficina:** representa los datos de cada una de las sucursales en las que los usuarios podrán atender y los clientes podrán tomar turnos.
- **Contador:** controla secuencia de turnos por cada uno de los servicios.
- **Service:** catálogo de servicios parametrizados en el sistema.
- **Turno:** almacena la información de cada uno de los turnos generados tanto de forma presencial como en línea.
- **Modulo:** contiene los módulos en los que cada usuario puede iniciar su atención, puede ser, por ejemplo: ventanilla 1.
- **Horario:** permite gestionar los horarios de atención de cada oficina para la toma de turnos en línea.

En la Tabla 13 se pueden observar la relación que existe entre la Colección 1 y la Colección 2 y que tipo de relación existen entre las dos entidades, esta información es útil para poder comprender de mejor manera como se manejará el sistema en lo respecto al almacenamiento de documentos de cada una de las colecciones.

Tabla 13

Relaciones entre colecciones

Colección 1	Colección 2	Tipo de Relación	Descripción
User	**Role**	Uno a muchos (1:*)	Un usuario puede tener múltiples roles.
User	**Oficina**	Uno a muchos (1:*)	Usuarios asignados a múltiples oficinas.
Turno	**Service**	Uno a uno (1:1)	Cada turno pertenece a un servicio.
Service	**Oficina**	Uno a uno (1:1)	Servicios vinculados a una oficina.
Horario	**Oficina**	Uno a muchos (1:*)	Horarios compartidos entre oficinas.

Pantalla de Inicio de sesión (RF01)

La pantalla de inicio de sesión solicita y valida un usuario y contraseña correctos además de la oficina que tenga asignada el usuario para que se le permita el acceso al sistema, la contraseña es enmascarada en la pantalla por seguridad.

Pantalla de Bienvenida con datos de usuario (RF03)

En esta pantalla de inicio se permite visualizar el menú con las opciones autorizadas e información relevante como nombre y oficina en la que el usuario ha iniciado sesión.

Cifrado bcrypt para contraseñas (RNF04)

Se realiza el cifrado de las contraseñas utilizando bcrypt al registrar un usuario y almacenarlo en la base de datos, de esta forma las contraseñas se encuentran cifradas y es necesario descifrarlas al validar cada uno de los inicios de sesión.

Control de acceso RBAC (RNF05)

Se desarrolló un middleware para la verificación de roles en node.js, esta funcionalidad permite evaluar un determinado rol en comparación a los que posee asignados el usuario, cada una de las peticiones a las diferentes rutas del API son filtradas por este middleware.

Autenticación JWT (RNF07)

Se proporciona al usuario un Jason Web Token (JWT) cuando su inicio de sesión es exitoso, este token le permite el acceso a cualquier petición adicional del API, se crea un middleware para verificar la validez del token recibido en las peticiones del usuario a las diferentes rutas.

Inicio Sprint 2: Gestión de Usuarios

Para cada uno de los requerimientos se pueden revisar detalles en el respectivo identificador del requerimiento en el desarrollo del sistema del ANEXO C.

Cambio de contraseña con validación (RF02)

En este requerimiento se valida la seguridad de la contraseña en el cambio de contraseña cuando se reinicia la contraseña o en el caso de que sea el primer ingreso.

Gestión CRUD de usuarios/roles (RF10)

En este requerimiento se desarrolla la pantalla de administración de usuarios, aquí se tienen todas las opciones de mantenimiento que componen el CRUD (creación, visualización, actualización y eliminación). Asimismo, se autorizan la adición y eliminación de roles para cada usuario a fin de permitir o restringir el acceso a diversas funciones.

Reinicio de contraseña temporal (RF25)

La pantalla de administración de usuarios permite el reinicio de las contraseñas de los usuarios previa confirmación, lo que genera el envío de un email con una contraseña temporal al correo electrónico registrado del usuario.

Arquitectura modular (RNF16)

En relación con este requerimiento en la codificación del sistema cada módulo de la aplicación se encuentra configurado y codificado de manera independiente para un fácil mantenimiento y escalabilidad.

Inicio Sprint 3: Turnos Básicos

Para cada uno de los requerimientos se pueden revisar detalles en el respectivo identificador del requerimiento en el desarrollo del sistema del ANEXO C.

Toma de turno presencial (RF04)

Este desarrollo permite la toma de turno de forma presencial para el usuario, la pantalla distribuye automáticamente de la mejor manera posible los servicios que se encuentren parametrizados en el sistema.

Visualización de turnos pendientes (RF05)

En esta pantalla el operador puede visualizar los turnos que tiene pendientes por atender en el módulo que inició su atención, con el fin de conocer cuántos turnos más es posible llamar en el sistema.

Listado de turnos en tiempo real (RF08)

En este requerimiento se permite que los clientes visualicen el turno en el que se encuentra en atención, y los turnos que serán próximos a atenderse, en los diferentes módulos y servicios.

Inicio Sprint 4: Turnos en Línea

Para cada uno de los requerimientos se pueden revisar detalles en el respectivo identificador del requerimiento en el desarrollo del sistema del ANEXO C.

Solicitud de identidad turno en línea y OTP (RF07)

Al tomar un turno en línea el sistema realiza el envío de un código OTP (Contraseña de un Solo Uso) al teléfono y email ingresados por el usuario, este código debe ser ingresado posteriormente para que el turno pueda obtenerse correctamente.

Validación de horarios (RF19)

El sistema permite seleccionar las oficinas que se encuentren activas dentro de un horario parametrizado en el sistema, en el caso de que no exista ninguna oficina parametrizada en el horario el sistema no permitirá la visualización de ninguna oficina.

Email de confirmación (RF21)

Al terminar el proceso de toma de turno en línea por completo, el usuario recibirá una notificación indicando detalles relevantes, como número de turnos en espera y tiempo de espera estimados.

Diseño responsive (RNF10)

Los controles del sistema se ajustarán correctamente en base al espacio que tengan disponible en una determinada pantalla, esto permite la correcta visualización de los componentes en diferentes dispositivos.

Inicio Sprint 5: Gestión Operativa

Para cada uno de los requerimientos se pueden revisar detalles en el respectivo identificador del requerimiento en el desarrollo del sistema del ANEXO C.

Transferencia de turnos (RF06)

Se permite seleccionar un servicio y un módulo al cual transferir el turno que se ha llamado previamente para su atención, esto permite que un usuario sea redirigido a otro servicio o módulo sin necesidad de tomar otro turno.

Validación de servicio en atención (RF16)

El sistema valida si existe ya un usuario que esté ocupando el módulo del servicio en el que se intenta iniciar atención, si un usuario ya se encuentra atendiendo el módulo no será posible que otro usuario lo use.

Visualización mejorada de turnos (RF22)

El software presenta una alerta visual cuando los turnos pendientes se actualizan realizando un resaltado de estos, esto con la finalidad de que los usuarios perciban inmediatamente los cambios en la cola de atención, optimizando así la gestión de tiempos y reduciendo confusiones.

Inicio Sprint 6: Administración Avanzada

Para cada uno de los requerimientos se pueden revisar detalles en el respectivo identificador del requerimiento en el desarrollo del sistema del ANEXO C.

Gestión de oficinas y servicios (RF09)

El sistema permite el mantenimiento completo con todas las operaciones necesarias de las oficinas y servicios que pueden ser utilizados o seleccionados tanto por los usuarios y los clientes.

Configuración de horarios (RF11)

Es posible la configuración de los horarios crearlos, editarlos o eliminarlos cada horario tiene una hora de fin e inicio los cuales definen la hora en los que pueden ser o no utilizadas las oficinas que estén atadas al horario.

Horarios por días de semana (RF26)

Al crear o editar un horario este puede contener los días en los que estará en vigencia, los cuales se configuran por su número de orden siendo lunes el número 1 y el domingo el número 7.

Inicio Sprint 7: Reportes y Estadísticas

Para cada uno de los requerimientos se pueden revisar detalles en el respectivo identificador del requerimiento en el desarrollo del sistema del ANEXO C.

Actualización de turnos pendientes (RF14)

La pantalla permite visualizar los turnos pendientes en tiempo real, conforme se tomen o atiendan turnos se actualiza por medio de una conexión integral al socket de la oficina con la utilización de socket.io.

Tiempos de atención (RF15)

En la pantalla de atención el usuario tiene la opción de realizar la llamada de un turno de los que estén en cola, al momento de realizar la llamada se visualizará en pantalla un cronómetro que indicará el tiempo que a tomado de atención el turno, el cual se detendrá al momento de atender, no atender, o transferir el turno.

Inicio Sprint 8: Mejoras de Usuario

Para cada uno de los requerimientos se pueden revisar detalles en el respectivo identificador del requerimiento en el desarrollo del sistema del ANEXO C.

Notificaciones audiovisuales (RF14)

El sistema genera una alerta auditiva y visual cuando se realiza una actualización de atención o toma de turno nuevo, esto funciona como un llamado al cliente o socio para que pase al módulo que le corresponde.

Ticket con QR (RF18)

Al tomar un turno presencial se imprime físicamente un ticket con algunos datos relevantes del turno, entre los cuales se tiene un código QR el cual sirve como validación de la originalidad del ticket.

Inicio Sprint 9: Seguridad y Validaciones

Para cada uno de los requerimientos se pueden revisar detalles en el respectivo identificador del requerimiento en el desarrollo del sistema del ANEXO C.

CAPTCHA para acciones sospechosas (RF20)

Se utiliza la validación de un captcha en la toma de turno en línea con la finalidad de evitar que el sistema sea usado de manera no permitida por un bot, el captcha se activa al realizar la toma de 3 turnos seguidos.

Configuración de voz (RF23)

En el sistema se permite la parametrización de la voz con la cual será alertado el cliente al momento de ser su turno, esta voz puede ser configurada entre masculina y femenina por oficina.

Validación de identidad (RF24)

En este requerimiento se puede configurar el sistema para que solicite un número de identificación al momento de tomar un turno presencial, el ingreso de identificación se realiza a través de un teclado virtual al ser en una pantalla táctil.

API’s para integración (RNF03)

La interacción entre el backend y frontend se realiza por medio de varios endpoint de un API, lo cual permite una fácil integración y deja abierta la posibilidad de integrar el sistema con cualquier otro sistema sin importar la plataforma o lenguaje de desarrollo.

Inicio Sprint 10: Optimizaciones Finales

Para cada uno de los requerimientos se pueden revisar detalles en el respectivo identificador del requerimiento en el desarrollo del sistema del ANEXO C.

Indicadores visuales (RNF14)

Durante las acciones que tomen tiempo en ejecución o necesiten una conexión externa se despliega en pantalla un indicador de carga conocido como spinner, el cual indica al usuario que el sistema está realizando un acción u no le permite realizar otras acciones hasta que termine la ejecución del proceso.

CAPÍTULO V

PROPUESTA

5.1. TÍTULO

Marco tecnológico con el uso del Stack MEAN aplicado a la gestión de turnos en cooperativas de ahorro y crédito

5.2. OBJETIVO

Proponer un Marco tecnológico con el uso de Stack MEAN para desarrollar y desplegar de un sistema para la administración de Turnos en la Cooperativa de Ahorro y Crédito Tulcán Ltda.

5.3. DESARROLLO

5.3.1. Arquitectura de despliegue

El sistema se basa en una arquitectura moderna utilizando el Stack MEAN (MongoDB, Express.js, Angular, Node.js) junto con Socket.io para la comunicación en tiempo real. Esta arquitectura puede visualizarse en los detalles en donde se puede observar cómo interactúan entre sí cada uno de sus componentes. Seguidamente, se explica el funcionamiento de cada parte dentro del sistema. (Ver sección “D.2. Arquitectura de despliegue” del ANEXO D).

Cliente/Usuario (Angular)

El usuario interactúa con la interfaz desarrollada en Angular, que permite realizar solicitudes como la generación de turnos, la consulta del estado y la atención de clientes. Angular se encarga de enviar estas solicitudes al backend y mostrar los resultados en la interfaz de usuario.

Servidor Backend (Node.js y Express.js)

Cuando el cliente realiza una solicitud, Angular la envía al servidor desarrollado en Node.js con el framework Express.js. El backend procesa la solicitud y, si es necesario, consulta o actualiza la base de datos en MongoDB. Una vez obtenida la respuesta, el servidor la devuelve a Angular para ser presentada al usuario.

Base de Datos (MongoDB)

La base de datos MongoDB almacena toda la información del software, incluyendo turnos, módulos de atención y usuarios. Express.js se comunica con la base de datos para registrar, consultar y actualizar los datos de los turnos en función de la solicitud del usuario.

Comunicación en Tiempo Real (Socket.io)

Para optimizar la eficiencia y evitar recargas automáticas de la página, el sistema utiliza Socket.io, una biblioteca que permite la comunicación instantánea entre cliente y servidor. Cada vez que se genera un nuevo turno, se atiende un cliente o hay un cambio en el estado del sistema, Socket.io emite eventos que notifican automáticamente a todos los clientes conectados, cada socket tiene un identificador propio lo que permite la división de estos en base a la necesidad como puede ser uno por cada oficina.

Funcionamiento de la arquitectura

1. El usuario en Angular solicita un turno o realiza una acción.
2. Angular envía la solicitud al backend en Node.js a través de una API REST.
3. Express.js procesa la solicitud y consulta MongoDB si es necesario.
4. MongoDB devuelve la información y Express.js responde a Node.js.
5. Node.js envía la respuesta a Angular para actualizar la interfaz del usuario.
6. Socket.io transmite eventos en tiempo real para sincronizar la información en todos los clientes conectados.

Esta arquitectura permite un sistema ágil, escalable y con una experiencia de usuario fluida, optimizando el manejo de turnos y la comunicación en tiempo real, la arquitectura del stack MEAN.

5.3.2. Diagramas de flujo

Para la construcción de esta propuesta se han planteado los diagramas de flujo recomendados del sistema de gestión de turnos que podrían ser utilizados para la implementación de este o guía de otros, estos se han basado específicamente en los requerimientos presentados y recopilados por este estudio enfocado en la

Cooperativa de Ahorro y Crédito Tulcán Ltda. (Ver sección “D.1. Diagramas de flujo propuestos para el sistema” del ANEXO D).

5.3.3. Arquitectura de aplicación

Arquitectura del backend

La arquitectura del backend que se propone sigue un patrón MVC (Modelo-Vista-Controlador) modularizado en cinco carpetas principales, cada una con responsabilidades específicas para garantizar escalabilidad y mantenibilidad. La estructura y funcionalidad se detalla en la Tabla 14. (Ver sección “D.2. Arquitectura de aplicación backend” del ANEXO D).

Tabla 14

Arquitectura de aplicación del backend

Carpeta	Propósito	Contenido	Relaciones
/models	Define la estructura de datos y lógica de negocio.	Esquemas de MongoDB (ej: User.js, Turno.js), validaciones y middlewares.	Se conecta con **controllers** y **routes**.
/controllers	Gestiona la lógica de operaciones y respuestas API.	Clases/funciones (ej: authController.js, turnoController.js) que procesan peticiones.	Recibe datos de **routes** y manipula **models**.
/routes	Maneja el enrutamiento de endpoints.	Definición de URLs (ej: authRoutes.js, apiRoutes.js) y middlewares de seguridad.	Llama a **controllers** según la ruta solicitada.
/utils	Contiene utilidades reutilizables y servicios auxiliares.	Helpers (ej: database.js, jwt.js), configuraciones, constantes y lógica compartida.	Usada por **controllers**, **models** y **routes**.

Carpeta	Propósito	Contenido	Relaciones
/assets	Almacena recursos estáticos necesarios para el servidor.	Archivos de configuración, plantillas de correo, certificados SSL o imágenes.	Referenciada desde otros módulos según necesidad.

Flujo de Trabajo:

El proceso inicia cuando una solicitud HTTP llega a un endpoint definido en la carpeta routes/, donde el enrutamiento redirige la petición al controlador correspondiente en controllers/. Este controlador ejecuta la lógica de negocio, interactuando con los esquemas y operaciones de base de datos definidos en models/. Para tareas específicas como validación de datos, autenticación o procesamiento especializado, se invocan funciones auxiliares almacenadas en utils/. Finalmente, el controlador genera y devuelve una respuesta HTTP estructurada (como JSON o mensajes de error) al cliente, cerrando el ciclo de la petición.

Arquitectura del frontend

El frontend se estructura en cuatro carpetas principales (interfaces, guards, modules y services), siguiendo estándares Angular para garantizar modularidad y escalabilidad. Las interfaces definen tipos de datos compartidos, los guards protegen rutas, y los servicios gestionan comunicación HTTP. La carpeta modules organiza funcionalidades en módulos independientes (ej: autenticación, turnos), cada uno con sus subcarpetas components, interfaces y services para componentes visuales, tipos de datos y lógica específica. Esta jerarquía favorece la cohesión, reutilización de código y lazy loading, optimizando mantenimiento y rendimiento, se puede ver Tabla 15. (Ver sección “D.2. Arquitectura de aplicación de frontend” del ANEXO D).

Tabla 15

Arquitectura del frontend

Carpeta	Propósito	Contenido	Relación
/interfaces	Define tipos y estructuras de datos globales.	Interfaces TypeScript (ej: User.interface.ts).	Usadas en services, components y modules.
/guards	Protege rutas con lógica de autenticación/autorización.	Clases como AuthGuard.ts que implementan CanActivate.	Integrado en el enrutamiento de modules.
/services	Centraliza llamadas HTTP y gestión de estado global.	Servicios (ej: Api.service.ts) con métodos reutilizables.	Consumidos por components y otros services.
/modules	Agrupa funcionalidades en módulos independientes (lazy loading).	Subcarpetas por módulo (ej: auth/, turnos/).	Importa components, services e interfaces locales.
/modules/[nombre]/components	Contiene componentes visuales reutilizables.	Componentes (ej: turno-list.component.ts).	Usan services locales e interfaces.
/modules/[nombre]/interfaces	Define tipos específicos del módulo.	Interfaces (ej: TurnoModel.interface.ts) para estructuras de datos del módulo.	Exclusivas para el módulo.

Carpeta	Propósito	Contenido	Relación
/modules/[nombre]/services	Gestiona lógica de negocio y comunicación con APIs.	Servicios (ej: Turno.service.ts) con métodos para operaciones del módulo.	Consumidos por components del mismo módulo.

La arquitectura del frontend ofrece un flujo de trabajo eficiente y múltiples beneficios gracias a su diseño organizado. La modularidad se logra encapsulando cada funcionalidad (como autenticación o gestión de turnos) en módulos independientes dentro de /modules, lo que permite la carga diferida (lazy loading) y reduce el tamaño inicial del bundle, optimizando el rendimiento. La cohesión se garantiza agrupando componentes, servicios e interfaces específicos de cada módulo en sus respectivas subcarpetas, lo que facilita el mantenimiento y la localización de código. Además, la reutilización de servicios globales (en /services) e interfaces globales (en /interfaces) evita la duplicación de código y centraliza la lógica común. Por último, la seguridad se refuerza mediante guards que validan el acceso a las rutas antes de renderizar componentes, protegiendo así las áreas restringidas de la aplicación. Esta estructura no solo mejora la escalabilidad y el trabajo en equipo, sino que también asegura un código limpio y fácil de actualizar.

Se puede visualizar información técnica del backend en el ANEXO B en donde también se encuentra la url para el visualización completa del código fuente tanto del frontend como del backend del sistema.

5.4. FINALIZACIÓN Y ENTREGA DEL PRODUCTO

5.4.1. Despliegue del Sistema en Ambiente de Producción

El despliegue del sistema de gestión de turnos en un servidor requirió una serie de pasos técnicos y configuraciones específicas para garantizar su correcto funcionamiento. Este procedimiento abarca la instalación de dependencias, la configuración del servidor y la implementación de herramientas para monitorizar y gestionar eficientemente los procesos. A continuación, se describe el proceso

utilizado para implementar el sistema en un servidor, que será la base para futuras implementaciones.

5.4.2. Instalación y Configuración del Backend

Instalación de MongoDB y MongoCompass

Se realiza la búsqueda del instalador de MongoDB en base al sistema operativo destino en el que se vaya a trabajar, para este caso Windows x64 y se procede con la instalación de MongoDB y MongoCompass los cuales vienen integrados en el mismo instalador, se realiza la instalación por defecto en este caso, finalmente se realiza la creación de cada una de las colecciones y documentos que se han planteado en el diseño de la base de datos, la creación para este caso se la ha realizado en el entorno gráfico MongoCompass. (Ver sección “D.3. Despliegue en servidor” del ANEXO D).

Instalación de Node.js y PM2

Se realiza la instalación de Node.js en el servidor de producción, un entorno de ejecución necesario para el despliegue del backend del sistema. Una vez instalado Node.js se procede a verificar su correcta instalación consultando la versión instalada y finalmente se procedió a instalar PM2, un administrador de procesos para Node.js que permite gestionar y mantener en ejecución las aplicaciones de manera eficiente. La instalación de PM2 se realizó mediante la consola de Windows después de esto se verificó la versión instalada. (Ver sección “D.3. Despliegue en servidor” del ANEXO D).

5.4.3. Despliegue del Proyecto

Se realizó una copia del proyecto en el servidor, a continuación, se accedió a la ruta del proyecto para instalar las dependencias necesarias. Esto se logró ejecutando el comando npm install, el cual instala todas las dependencias definidas en el archivo package.json en la carpeta node_modules. (Ver sección “D.3. Despliegue en servidor” del ANEXO D).

Ejecución del Backend con PM2

Para iniciar el backend, se utilizó PM2 con un archivo de configuración (ecosystem.config.cjs) este archivo contiene variables de entorno dentro del proyecto, como la zona horaria, para evitar inconsistencias en el manejo de fechas y horas dentro del sistema, se procedió con el inicio de la aplicación con el comando pm2 start ecosystem.config.cjs y adicionalmente, se monitorearon los logs en tiempo real mediante el comando pm2 logs, lo que permitió identificar y solucionar posibles errores durante el inicio del servidor, finalmente se valida el correcto levantamiento de la aplicación en el log, comprobando la conexión exitosa a la base de datos la cual se encuentra en este caso en el mismo servidor. (Ver sección “D.3. Despliegue en servidor” del ANEXO D).

Despliegue del Frontend con Apache

1. Configuración del Servidor Web

Para el despliegue del frontend, desarrollado en Angular, se utilizó el servidor de aplicaciones web Apache. Este servidor se configuró para entregar los archivos estáticos generados en el proceso de construcción de la aplicación Angular; la versión y descarga se pueden ver en los detalles, Apache puede ser iniciado gráficamente usando el Apache Service Monitor.

La generación del bundle para producción se realiza con el comando ng build–prod si se presenta un error al ejecutar el comando relacionado con los tamaños del archivo, se deberá configurar el archivo angular.json aumentando los tamaños permitidos. Una vez realizada esta corrección si el bundle es generado correctamente se observará un mensaje exitoso y se habrá creado una carpeta con el nombre dist en nuestro proyecto.

Este proceso realizado anteriormente para la generación del build de producción fue generado para este caso para el proyecto de toma de turno en línea ya que para este caso se han realizado proyectos diferentes al requerirse que uno de ellos funcione en la intranet y otro sea expuesto al internet.

Es necesario que la carpeta dist, que contiene los archivos compilados, se coloque en el directorio raíz de Apache (htdocs). (Ver sección “D.3. Despliegue en servidor” del ANEXO D).

2. Manejo de Rutas en Angular

Para evitar problemas con las rutas en el servidor, se configuró el sistema de enrutamiento de Angular utilizando el hash (#) en las URLs. Esto se logró modificando el archivo app-routing.module.ts. Alternativamente, en caso de no utilizar el hash, se debe configurar el archivo app-routing.module.ts, posterior a esto se realiza el bundle de producción con una ruta base, es necesario des comentar algunas partes la configuración en el archivo httpd.conf de apache ubicado en la ruta C:/Apache24/conf/httpd.conf para este caso de configuración. Se necesita validar que en el archivo index.html en la ruta de acceso C:/Apache24/htdocs/TurnoPresencial/ exista cierta configuración de la ruta base, adicionalmente se debe configurar el archivo .htaccess que se encuentra en la ruta de acceso C:/Apache24/htdocs/TurnoPresencial si no existe se debe crearlo, por último, se configura el archivo httpd-vhosts.conf para el caso de este proyecto. (Ver sección “D.3. Despliegue en servidor” del ANEXO D).

Es necesario reiniciar el servidor de aplicaciones de Apache en cualquiera de los dos casos de configuración.

5.5. PRUEBAS Y VERIFICACIÓN

Tras finalizar la configuración, se llevaron a cabo evaluaciones para confirmar el adecuado funcionamiento del software en backend y frontend. Se monitorearon las peticiones recibidas por el servidor mediante los logs de PM2, y se validó que las interfaces de usuario (tanto para la toma de turnos presenciales como en línea) respondieran de manera adecuada. (Ver sección “D.4. Pruebas y Verificación” del ANEXO D).

5.5.1. Evaluación y validación por parte de los usuarios

Como parte fundamental en el proceso de verificación, se realizaron pruebas funcionales en sesiones de trabajo presenciales con los usuarios finales del sistema, como asistentes de negocio, atención al cliente, ventanilla, etc. Estas pruebas funcionales permitieron, en un entorno real, validar los requisitos funcionales y los requisitos no funcionales caracterizados durante la fase de levantamiento.

Las sesiones se documentaron mediante actas de prueba del desarrollo firmadas por los asistentes a la sesión, donde se recogieron los módulos evaluados, el éxito o fallo de cada prueba y observaciones puntuales en caso de propuestas de mejora. Las pruebas confirmaron tanto el sistema de gestión de los turnos presenciales como el sistema de gestión de los turnos en línea.

Los resultados son conformes a los objetivos del proyecto; por lo que se delimita que todos los módulos se verificaron con resultado exitoso (se recogieron mejoras factibles posteriores en función de las propuestas de los usuarios). Estas evidencias están recogidas en los ANEXOS F y G.

5.6. FINALIZACIÓN Y ENTREGA DEL PRODUCTO

En los momentos finales antes de la entrega y obtención de resultados del sistema de gestión de turnos, se llevó a cabo una revisión meticulosa con el personal administrativo y técnico de la Cooperativa de Ahorro y Crédito Tulcán Ltda., para verificar que el diseño y funcionalidad del sistema cumplían con los requisitos establecidos en la etapa previa. Así, se aseguraría que el producto final fuese confiable, operativo y apto para su producción.

Durante la entrega, se generaron todos los archivos del sistema, incluyendo el código fuente, la base de datos en formato MongoDB, las credenciales del administrador, archivos de configuración en JSON, documentación técnica, capturas de pantalla que muestran el funcionamiento correcto de las principales interfaces, etc.

Finalmente, se firmó un acta de entrega a nivel técnico y una a nivel funcional por los encargados del proyecto dentro de la institución, como se puede observar en ANEXO H y I respectivamente, y respaldado por las actas de pruebas funcionales en los ANEXOS F y G, donde los usuarios verificaron el cumplimiento de los requerimientos y emitieron observaciones durante el proceso de desarrollo, esto como prueba formal de la auditoría de satisfacción de cumplimiento de los requisitos y objetivos definidos en la investigación, siendo testigos de la viabilidad o confirmación de su ejecución en el entorno productivo de la cooperativa.

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

La arquitectura modular del stack MEAN permitió una incorporación sencilla en la infraestructura preexistente, a la vez que su arquitectura basada en JavaScript facilita una mejor escritura y mantenimiento.

La posibilidad de manejar la transmisión de datos en tiempo real gracias a Socket.io mejoró la experiencia de usuario, proporcionando actualizaciones instantáneas desde la pantalla de turnos, sin que el usuario tuviera que refrescar manualmente la aplicación entre actualizaciones.

La aplicación de la metodología SCRUM en la creación del sistema facilitó una gestión adaptable y flexible del proyecto. La segmentación en sprints permitió un entregable progresivo de las funcionalidades del sistema, a la vez que las retrospectivas periódicas permitieron detectar y subsanar los problemas de manera temprana.

El sistema de información creado se ajusta a la normativa legal y regulatoria del sector financiero ecuatoriano, incluyendo: la Ley de Protección de Datos, la Ley de Cibercrimen, el Acuerdo de la Junta de Política y Regulación Monetaria y Financiera, el Código Orgánico Monetario y Financiero, entre otros. Utilizando seguridades como el cifrado con bcrypt de las contraseñas o el uso de JWT para la autenticación, su arquitectura ratificó su propio grado de tolerancia a la generación de reportes para auditorías, tanto internas como externas.

El marco tecnológico propuesto no solo responde a las necesidades específicas de la Cooperativa de Ahorro y Crédito Tulcán Ltda., sino que, al estar basado en arquitecturas modernas, también puede aplicarse de manera general a otras cooperativas de ahorro y crédito en el Ecuador.

Recomendaciones

Los entrenamientos de formación aplicables no solo tienen que incluir el uso básico de la plataforma, sino también aspectos más avanzados como la generación de reportes o la gestión de incidencias, y tener un manual de procedimientos que esté actualizado y que sirva de guía rápida para los usuarios. La cooperativa podría formar alianzas con universidades o centros de formación para certificar a sus colaboradores el manejo del stack MEAN.

A mediano plazo, adaptar: inteligencia artificial para poder predecir los picos de demanda, dispositivos biométricos para aumentar la seguridad, chatbots para la atención automatizada de consultas recurrentes y una aplicación móvil nativa para complementar la versión web.

Documentar: diagramas de arquitectura, manuales de instalación, códigos de error comunes y guías para desarrolladores; esta documentación deberá estar alojada en un repositorio accesible, por ejemplo, un GitHub Wiki o un Confluence y deberá actualizarse en cada nueva versión del software.

REFERENCIAS

Abuchar Porras, A. (2023). *Metodologías ágiles para el desarrollo de software.* Bogotá: Editorial Universidad Distrital Francisco José de Caldas.

Agarwal, S., Yadav, R., Narula, T., & Agarwal, A. (2024). MEAN STACK. *Journal of Analysis and Computation (JAC), 18(2),* 352-361. Obtenido de https://ijaconline.com/wp-content/uploads/2024/08/NG.2-33.pdf

Aguirre Barrera, J., & Aguirre Barrera, S. (2020). Metodologías para el desarrollo de Proyectos. Obtenido de https://repository.unicatolica.edu.co/bitstream/handle/20.500.12237/2037/ART%c3%8dCULO_METODOLOG%c3%8dAS_PARA_DESARROLLO_PROYECTOS.pdf?sequence=1&isAllowed=y

Alamilla Hernández, L., Pérez Romero, V., Sosa González, S., & Valentín Rodríguez, J. (2021). Arquitectura REST para el desarrollo de aplicaciones web empresariales. *Revista Electrónica sobre Ciencia, Tecnología y Sociedad, 8*(15). Obtenido de https://www.ctes.org.mx/index.php/ctes/article/view/748/909

Armijos Ortega, L., Vélez Macas, C., & Lojan Cueva, E. (2025). Estudio de la adopción de metodologías ágiles en proyectos de desarrollo de software en la región 7 del Ecuador. *Revista Espacios, 46*(10), 73-84. Obtenido de https://www.revistaespacios.com

Arsaute, A., Zorzan, F., Daniele, M., González, A., & Frutos, M. (2018). Generación automática de API REST a partir de API Java, basada en transformación de Modelos (MDD). *XX Workshop de Investigadores en Ciencias de la Computación*, 629-634. Obtenido de http://sedici.unlp.edu.ar/handle/10915/67777

Asamblea Nacional del Ecuador. (2002). *Ley de Comercio Electrónico, Firmas Electrónicas y Mensajes de Datos. Registro Oficial No. 557.* Ecuador. Obtenido de https://www.regulacion.gob.ec

Basantes Espinoza, G. P. (2020). *Implementación de un sistema de asignación de turnos para la atención a clientes que se acercan al Hall de servicios para una institución financiera* (Tesis de Maestría). Escuela Superior Politécnica del Litoral (ESPOL), Guayaquil, Ecuador

Canós, J., Letelier, P., & Penadés, M. (s.f.). Metodologías Ágiles en el Desarrollo de Software. *Universidad Politécnica de Valencia*. Obtenido de http://issi.dsic.upv.es/archives/f-1069167248521/actas.pdf

Castillo Estrada, C., Cancino Villatoro, K., Benavides García, V., & De la Cruz Vázquez, A. (2022). Diseño de un Sistema web para el control de Curriculum Vitae Electrónico de personal docente basado en una arquitectura orientada a servicios (API REST). *RITI Journal, 10*(20), 28-42. DOI:https://doi.org/10.36825/RITI.10.20.003

Chavez Ponce et al. (2022). Revisión de modelos que integren Design Thinking en metodologías de Desarrollo Ágil. *Revista Innovación y Software, 3*(1), 47-57. Obtenido de https://revistas.ulasalle.edu.pe/innosoft/article/view/52/57

Choto Maza, J., Avila, D., & Avila Pesantez, L. (2020). Desarrollo de una aplicación móvil utilizando el framework MEAN Stack e IONIC: Un estudio de caso en una compañía de transporte. *Ecuadorian Science Journal, 4*(2), 37-42. DOI:https://doi.org/10.46480/esj.4.2.74

Código Orgánico Monetario y Financiero. (11 de Febrero de 2022). *Segundo Suplemento del Registro Oficial No.332*. Obtenido de https://biblioteca.defensoria.gob.ec/bitstream/37000/3399/1/C%c3%b3digo%20Org%c3%a1nico%20Monetario%20y%20Financiero.pdf

Constitución de la República del Ecuador. (20 de Octubre de 2008). *Registro Oficial 449*. Obtenido de https://www.defensa.gob.ec/wp-content/uploads/downloads/2021/02/Constitucion-de-la-Republica-del-Ecuador_act_ene-2021.pdf

da Rocha Nascimento, M. A. (2021). Discrete event simulation applied to single queue management: a case study at a bank agency. *Simulation Modelling Practice and Theory*, 108.

Deemer, P., Benefield, G., Larman, C., & Vodde, B. (2012). *Scrum Primer.* Obtenido de Una introducción básica a la teoría y práctica de Scrum: https://scrumprimer.org/primers/es_scrumprimer20.pdf

Flores Zavala, G., Villegas Cayllahua, F., & Napán Yactayo, A. (2021). Calidad de servicio y su relación con la fidelización de los clientes. *Revista Arbitrada Interdisciplinaria KOINONIA, 6*(1), 200-221. DOI:https://doi.org/10.35381/r.k.v6i1.1225

García, M., & Proaño, J. (2020). Mejora de la atención al cliente a través de la tecnología en instituciones financieras ecuatorianas. *Revista de Tecnología, Innovación y Negocios, 5*(1), 47-60.

García, M., Pérez, A., & Rodríguez, L. (2023). Tecnologías avanzadas en la gestión de turnos laborales. *Journal of Operational Management, 15*(2), 134-150. DOI:https://doi.org/10.1016/j.jom.2023.02.004

Giler Araujo, J. J., Alvarado Cagua, K. J., Yela Burgos, R. T., Choez Muñiz, F. V., & García Salazar, J. A. (2024). *Impacto de la digitalización en el sector bancario ecuatoriano: transformaciones, desafíos y oportunidades*. *LATAM Revista Latinoamericana de Ciencias Sociales y Humanidades, 5*(5), 936–953. DOI:https://doi.org/10.56712/latam.v5i5.2658

Gimba, U. et al. (2020). Queue monitoring system for bank. *Dutse Journal of Pure and Applied Sciences, 6*(2), 45-62.

Gómez, L., & Pérez , A. (2022). Estrategias de mejora en la atención al cliente mediante la gestión de turnos. *Revista de Administración Financiera, 15*(2), 45-58. DOI:https://doi.org/10.1234/raf.v15i2.12345

Sánchez Aristi, Á., Mendieta Aragón, A., & Arguedas Sanz, R. (2023). *El sector financiero en la era digital: Datos, digitalización y transformación tecnológica*. Editorial Tirant lo Blanch.

Gowda, P. G. (2022). Implementing Authentication and session management in an Angular JS single-page application. European Journal of Advances in Engineering and Technology, 9(7), 81-86. Obtenido de https://www.researchgate.net/profile/Priyanka-Gowda-Ashwath-Narayana-2/publication/Implementing

Heredia, J., & Sailema, G. (2018). Comparative Analysis for Web Applications Based on REST Services: MEAN Stack and Java EE Stack. *KnE Engineering, 3*(9), 82. doi:https://doi.org/10.18502/keg.v3i9.3647

IEEE. (1998). *IEEE Recommended Practice for Software Requirements Specifications* (IEEE Std 830-1998). Institute of Electrical and Electronics Engineers. https://doi.org/10.1109/IEEESTD.1998.88286

Izquierdo Espinoza, J., & Anastacio Vallejos, C. (2021). Calidad De Servicio En Instituciones Privadas Y Públicas: Revisión Sistemática. *TZHOECOEN, 13*(2), 84-93. DOI:https://doi.org/10.26495/tzh.v13i2.2002

Khan, Wisal et al. (2023). SQL and NoSQL Database Software Architecture Performance Analysis and Assessments—A Systematic Literature Review. *Big Data Cogn. Comput., 7*(2). DOI:https://doi.org/10.3390/bdcc7020097

Le, D. A. (2023). *E-Commercial Full Stack Web Application Development.* Obtenido de https://www.theseus.fi/bitstream/handle/10024/802860/E-Commercial%20Full%20Stack%20Web%20Application%20Development.pdf?sequence=2

Ley Orgánica de Defensa del Consumidor. (2000). *Ley No. 2000-21*. Obtenido de https://www.gob.ec/sites/default/files/regulations/2018-09/Documento_Ley-Organica-Defensa-Consumidor.pdf

López, R., & Martínez, J. (2023). Tecnologías emergentes en la gestión de filas para instituciones financieras. *Journal of Financial Services, 28*(3), 99-115. DOI:https://doi.org/10.1234/jfs.v28i3.67890

Maida, E. G., & Pacienzia, J. (Diciembre de 2015). *Metodologías de desarrollo de software.* Obtenido de Biblioteca Digital de La Universidad Católica Argentina: https://repositorio.uca.edu.ar/bitstream/123456789/522/1/metodologias-desarrollo-software.pdf

Martínez Padua, J., Quitian Monroy, J., & Castiblanco Jiménez, I. (2022). Caracterización y comparación de metodologías ágiles y tradicionales de desarrollo de producto. *Ciencia e Ingeniería Neogranadina, 32*(2), 9-26. DOI:https://doi.org/10.18359/rcin.5168

Méndez, F. (2023). La relación entre la satisfacción del cliente y la gestión de turnos en el sector financiero. *Revista de Marketing y Finanzas, 22*(4), 120-134. DOI:https://doi.org/10.1234/rmf.v22i4.34567

MongoDB Inc. (s. f.). *What is the MEAN stack?* https://www.mongodb.com /resources/languages/mean-stack

Miranda Cruz, M. et al. (2021). La calidad de los servicios y la satisfacción del cliente, estrategias del marketing digital. Caso de estudio hacienda turística rancho los emilio´s. Alausí. *Dominio de las Ciencias*, 1430-1446.

Mohanish, Bawane et al. (2022). A Review on Technologies used in MERN stack. *International Journal for Research in Applied Science & Engineering Technology (IJRASET), 10*(1), 478-489. DOI:https://doi.org/10.22214/ijraset.2022.39868

Molina Montero, B., Vite Cevallos, H., & Davila Cuesta, J. (2018). Metodologías ágiles frente a las tradicionales en el proceso de desarrollo de software. *Espirales revista multidisciplinaria de investigación, 2*(17), 118. DOI:https://doi.org/10.31876/re.v2i17.269

Molina Montero, B., Vite Cevallos, H., & Dávila Cuesta, J. (2018). Metodologías ágiles frente a las tradicionales en el proceso de desarrollo de software. *Espirales Revista Multidisciplinaria De investigación*, 2(17). DOI:https://doi.org/10.31876/re.v2i17.269

Morales Carrillo, J., Cedeño Valarezo, L., Cajape Bravo, J., & Ormaza Calderón, J. (2022). Metodologías de desarrollo de software y su ámbito de aplicación: Una revisión sistemática. *Iberian Journal of Information Systems and Technologies*(47), 29-45. Obtenido de https://www.risti.xyz/issues/ristie47.pdf

Nevalainen, T. (2018). *MEAN Software Stack.* Obtenido de https://www.theseus.fi/bitstream/handle/10024/154731/Nevalainen _Toni.pdf?sequence=1&isAllowed=y

Ngoc Le. (2023). *Web Application for Searching Local Events.* Obtenido de https://www.theseus.fi/bitstream/handle/10024/800405/NgocLe_thesis_final.pdf?sequence=2

Olaiya, O et al. (2024). The impact of big data analytics on financial risk management. *International Journal of Financial Studies, 12*(2), 1313. DOI:https://dx.doi.org/10.30574/ijsra.2024.12.2.1313

Otzen , T., & Manterola, C. (2017). Técnicas de Muestreo sobre una Población a Estudio. *International Journal of Morphology, 35*(1), 227-232. DOI:https://doi.org/10.4067/S0717-95022017000100037

Palacios Vega, P., Álvarez Gavilanes, J., & Ramírez Valarezo, C. (2021). Gestión de calidad del proceso de Atención al Usuario. *CIENCIAMATRIA. Revista Interdisciplinaria de Humanidades, Educación, Ciencia y Tecnología, 7*(12), 67-96. doi:DOI 10.35381/cm.v7i12.421

Prati , J., & Manjunath, C. (2021). Comparative analysis of MEAN stack and MERN stack. *Journal of Emerging Technologies and Innovative Research (JETIR), 8*(5), 614-620. Obtenido de https://www.jetir.org/papers/JETIR2105881.pdf

Pratyusha, G. et al. (2023). DEVELOPMENT OF A COLLEGE PLACEMENT WEB SITE USING THE MEAN STACK. *Dogo Rangsang Research Journal, 13*(20), 31-35. DOI:10.36893.DRSR.2023.V13I04.0031-0035

Pressman, R. S., & Maxim, B. R. (2020). *Software Engineering: A Practitioner's Approach* (9th ed.). McGraw-Hill Education.

Quezada Torres, W., & Chamba Mendez, C. (2023). Sistema CRM para la gestión de atención al cliente en las cooperativas de ahorro y crédito ecuatorianas. *Universidad y Sociedad, 15*(3), 149-155. Obtenido de http://scielo.sld.cu/scielo.php?script=sci_arttext&pid= S2218- 36202023000300149&lng=es&nrm=iso>. Epub 30-Jun-2023. ISSN 2218-3620.

Ramírez, P., & Torres , S. (2022). Impacto de la distribución de turnos en la calidad del servicio bancario. *Gestión Bancaria, 17*(1), 72-85. DOI:https://doi.org/10.1234/gb.v17i1.56789

Rana et al. (2024). An efficient artificial intelligence (AI) & Internet of things (IoT's) based MEAN stack technology applications. *Bulletin of Business and Economics, 13*(2), 200-206. DOI:https://doi.org/10.61506/01.00316

Reinoso Coello, M. (2019). *Modelo de gestión de proceso apoyado en la tecnología Stack Mean (Tesis de maestría no publicada).* Pontificia Universidad Católica del Ecuador Sede Ambato, Ambato.

Rivas, C., Corona, V., Gutiérrez, J., & Hernández, L. (Diciembre de 2015). Metodologías actuales de desarrollo de software. *Revista Tecnología e Innovación, 2*(5), 980-986.

Rodriguez , E. (2019). Análisis de la calidad del servicio en instituciones financieras en el norte de Ecuador. *Revista de Ciencias Administrativas y Financieras, 7*(2), 78-92.

Romani Alejo et al. (2023). *La eficiencia de la calidad de servicio al cliente de las entidades financieras.* Chiclayo, Perú: Proyecto CIDE Editorial. Obtenido de https://repositorio.cidecuador.org/bitstream/ 123456789/2392/1/2Libro%20La%20Eficiencia.VF_24_52023.pdf

Romero Hinojoza, Á., Castillo Jaramillo, M., & León Prieto, L. (2022). Metodologías ágiles como herramienta tecnopedagógica: ventajas y desventajas. *Ciencia Latina Revista Científica Multidisciplinar, 6*(3), 4223-4240. DOI:https://doi.org/10.37811/cl_rcm.v6i3.2552

Saeed, I et al. (2022). Towards Examining the Link Between Workplace Spirituality and Workforce Agility: Exploring Higher Educational Institutions. *Psychology Research and Behavior Management, 15*(1). DOI:https://dx.doi.org/10.2147/PRBM.S344651

Safdar, K. et al. (2021). An optimized queue management system to improve patient flow in the absence of appointment system. *Annals of Operations Research, 298*(1), 321-345.

Sanchit Aggarwal, & Jyoti Verma. (2018). Comparative analysis of MEAN stack and MERN stack. *International Journal of Recent Research Aspects, 5*, 127-132. Obtenido de https://www.ijrra.net/Vol5issue1/IJRRA-05-01-26.pdf

Sarwar, H. et al. (2023). An Efficient E-Commerce Web Platform Based on Deep Integration of MEAN Stack Technologies. *Bulletin of Business and Economics, 12*(4), 447-453. DOI:https://doi.org/10.61506/01.00147

Sayago Heredia, J., & Revelo Bautista, F. (2022). Aplicaciones web modernas con stack MEAN: Un caso de estudio. *Revista Killkana Técnica, 6*(1), 31-41. DOI: https://doi.org/10.26871/killkanatecnica.v6i1.989

Schwaber, K., & Sutherland, J. (2020). *La Guía de Scrum: Las reglas del juego*. Scrum.org. https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Spanish-Latin-South-American.pdf

Sharma, R., Bajpai, A., Maheshwari, A., Sharma, A., & Gupta, G. (2022). Transforming Inventory Management System using MEAN Stack. *International Research Journal of Engineering and Technology (IRJET), 9*(3), 1386-1392. Obtenido de https://d1wqtxts1xzle7.cloudfront.net/89481460/IRJET_V9I3256-libre.pdf

Smith, T., & Johnson, R. (2022). Optimización de costos laborales a través de la gestión de turnos. *International Journal of Human Resource Management, 33*(4), 567-582. DOI:https://doi.org/10.1080/09585192.2022.1123456

Sotomayor, A., & Paz, J. (2021). Experiencia del cliente en instituciones financieras en Ecuador: Un estudio exploratorio. *Revista de Estudios Empresariales, 5*(1), 112-128.

Tenesaca Machúcala, B., & Rodríguez Pillaga, R. (2022). Calidad de servicio y satisfacción del usuario en instituciones financieras. *CIENCIAMATRIA. Revista Interdisciplinaria de Humanidades, Educación, Ciencia y Tecnología, VIII*(2), 116-135. doi:DOI 10.35381/cm.v8i2.701

Valpadasu Hema et al. (2020). *Scrum: An Effective Software Development Agile Tool.* doi:10.1088/1757-899X/981/2/022060

Vera León, W. A. (2023). *Sistema web y móvil para la gestión de turnos en instituciones financieras con colas virtuales y reservas en línea* (Tesis de Maestría). Universidad Continental, Perú. Recuperado de https://repositorio.continental.edu.pe/bitstream/20.500.12394/12440/2/IV_PG_MGP_TI_Vera_Leon_2023.pdf

Vyas, R. (2022). Comparative Analysis on Front-End Frameworks for Web Applications. *International Journal for Research in Applied Science*

& Engineering Technology (IJRASET), 10(7), 298-307. DOI:https://doi.org/10.22214/ijraset.2022.45260

Weber, N. (2022). *Evaluation and Comparison of Full-Stack JavaScript Technologies.* Obtenido de https://opus.hs-offenburg.de/frontdoor/index/index/searchtype/authorsearch/ author/Nadine+Weber/docId/6125/start/0/rows/10

ANEXOS

ANEXO A. Especificación de requisitos - estándar IEEE 830-1998

1. Introducción

1.1. Propósito

El presente documento sirve como soporte para definir cuáles son los requerimientos funcionales y no funcionales del sistema que ha sido desarrollado en este trabajo de investigación, el software de gestión de turnos para la Cooperativa de Ahorro y Crédito Tulcán Ltda., se desarrolló basándose en la tecnología del stack MEAN (MongoDB, Express.js, Angular, Node.js), este documento puede ser utilizado como una guía para los administradores o usuarios que pueden ser técnicos o funcionales y así como las partes interesadas.

1.2. Alcance

La presente especificación de requisitos se ha realizado para los usuarios del sistema directos como el personal de atención al cliente e indirectos como los clientes de la Cooperativa de Ahorro y Crédito Tulcán Ltda., con lo que se busca continuar mejorando el ambiente tecnológico y automatizaciones dentro de la institución, que tiene como principal objetivo la mejora en tiempos operativos y administrativos de los diferentes procesos que se llevan a cabo dentro de la institución.

1.3. Personal involucrado

A continuación, se detalla al personal que estará involucrado en la creación y definición del documento:

Nombre	**Iván Darío Rojas Rojas**
Rol	Analista, diseñador y programador
Categoría Profesional	Analista de Tecnologías de la Información
Información de contacto	ivan.rojas@cooptulcan.fin.ec

Nombre	**Gabriel Romeo Gordon Benitez**
Rol	Definición de aspectos tecnológicos
Categoría Profesional	Jefe de Tecnologías de la Información
Información de contacto	gabriel.gordon@cooptulcan.fin.ec

Nombre	**Adriana Paola Morales Cruz**
Rol	Definiciones en aspectos de atención al cliente
Categoría Profesional	Oficial de Atención al Cliente
Información de contacto	paola.morales@cooptulcan.fin.ec

Nombre	**Pablo Eduardo Hernandez Rosero**
Rol	Definiciones de aspectos visuales del sistema
Categoría Profesional	Jefe de Marketing
Información de contacto	pablo.hernandez@cooptulcan.fin.ec

1.4. Definiciones, acrónimos y abreviaturas

Se deberá definirán los términos. Acrónimos y abreviaturas que se utilizaran en el ERS.

Acrónimo / Término	Definición
MEAN Stack	Conjunto de tecnologías (MongoDB, Express.js, Angular, Node.js) para desarrollo web.
OTP	Contraseña de un solo uso.
RBAC	Control de Acceso Basado en Roles.
JWT	JSON Web Token para autenticación.
API	Interfaz de Programación de Aplicaciones.

1.5. Referencias

Título del Documento	Referencia

1.6. Resumen

Este documento se organiza en tres secciones: una introducción con una visión general de los requisitos, una descripción general del sistema que resume sus funciones, datos y restricciones, y una última sección donde se detallan los requisitos específicos que el sistema debe cumplir.

2. Descripción general

2.1. Perspectiva del producto

Los casos de uso descritos aquí se han creado según los requisitos obtenidos de los usuarios de la Cooperativa de Ahorro y Crédito Tulcán. Estos casos de uso explican las interacciones clave entre los usuarios y el sistema, tratando los procesos que serán esenciales para el buen funcionamiento del sistema en el entorno operativo de la cooperativa. Cada caso de uso refleja las necesidades específicas de los usuarios, asegurando que el sistema satisfaga las expectativas tanto del lado técnico como funcional.

2.2. Características de los usuarios

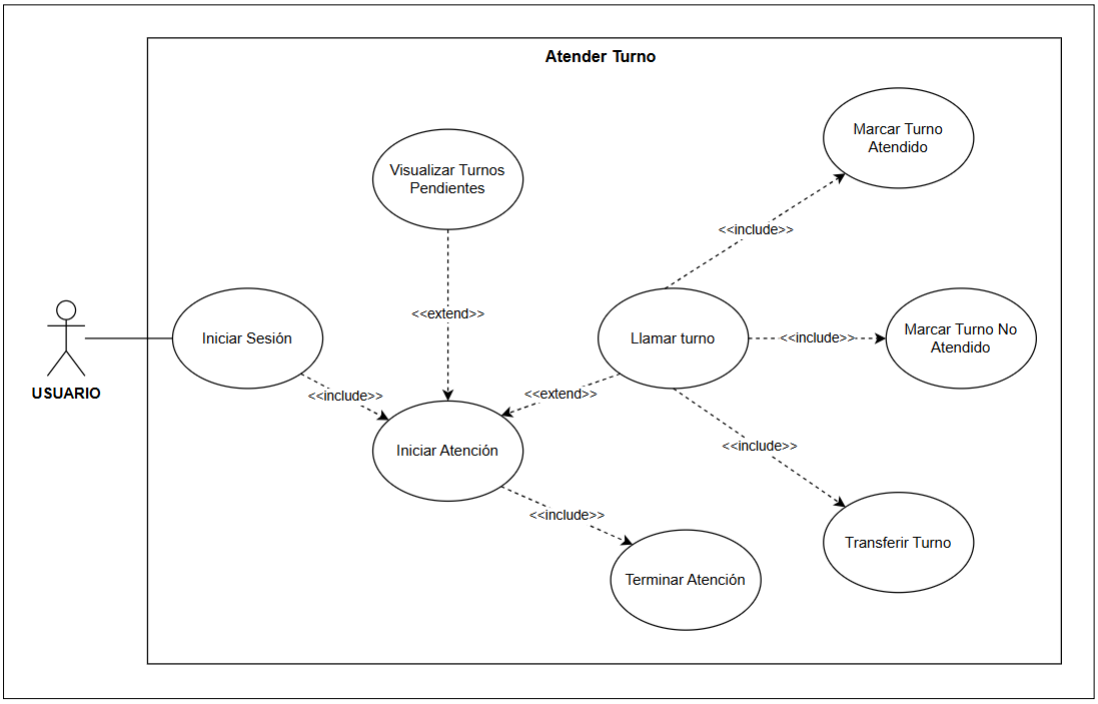
Tipo de usuario	**Operador / Usuario de atención al cliente**
Formación	Capacitación básica en computación y atención al cliente
Actividades	Atención de turnos, visualización de lista de espera, marcar turnos como atendidos o no atendidos, transferir turnos entre módulos.

Tipo de usuario	**Cliente**
Formación	No requerida
Actividades	Solicita turnos presencialmente o en línea, recibe notificaciones, espera ser llamado y puede calificar la atención recibida.

Tipo de usuario	Administrador
Formación	Técnico o profesional en Informática o Sistemas
Actividades	Control y manejo general del sistema: creación de usuarios, configuración de oficinas, horarios, servicios, módulos y generación de reportes.

2.3. Funcionalidad del producto

Caso de uso Atender Turno



Caso de Uso: Atender Turno

Descripción: Este caso de uso detalla cómo un usuario de atención al cliente maneja turnos.

Actores: Usuario de atención al cliente, Sistema de Turnos.

Precondiciones: El operador debe estar logueado en el sistema y debe haber comenzado la atención en un módulo y servicio.

Flujo Principal:

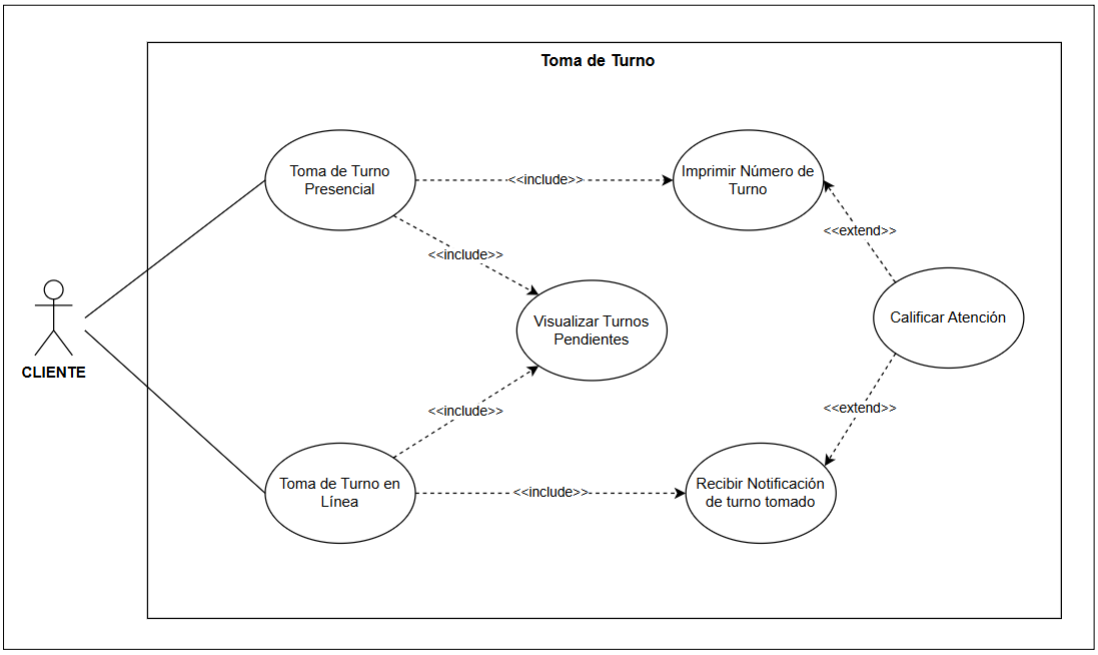
- El usuario inicia sesión en el sistema.
- El usuario inicia la atención en un servicio y módulo específicos.

- El usuario visualiza los turnos en cola.
- El usuario llama un turno para ser atendido.
- El sistema marca el turno como "atendido".
- El usuario termina la atención cuando sea necesario.

Flujos Alternativos:

- Si el turno no puede ser atendido, el operador marca el turno como "no atendido".
- Si el turno debe ser transferido, el operador transfiere el turno a otro módulo o agente.

Caso de uso Toma de Turno



Caso de Uso Toma de Turno Presencial y en Línea

Descripción: Este caso de uso detalla la forma mediante el cual un cliente toma un turno, ya sea de manera presencial (en la oficina) o en línea (a través del sistema web). El sistema genera un número de turno y notifica al cliente, quien espera a ser atendido.

Actores: Cliente, Sistema de Turnos

Precondiciones:

Para la **toma de turno presencial**:

- El cliente debe estar físicamente en la oficina.
- La oficina debe tener disponibilidad del servicio deseado.

Para la toma de turno en línea:

- El cliente debe tener acceso a internet.
- El cliente debe ingresar los datos solicitados.
- El servicio solicitado debe estar disponible para la oficina y horario seleccionados.

Flujo Principal:

Selección del Servicio: El cliente selecciona el servicio que requiere.

- **Presencial:** El cliente selecciona el servicio en una pantalla o mediante interacción con un operador.
- **En Línea:** El cliente selecciona el servicio a través de una interfaz digital e ingresará todos los datos solicitados en la página web.

Generación del Número de Turno: El sistema genera un número de turno único para el cliente.

- **Presencial:** El número de turno se imprime en un ticket físico.
- **En Línea:** El número de turno se muestra en la pantalla junto con información adicional del turno y se envía una notificación al cliente (por correo electrónico).

Notificación al Cliente: El cliente recibe el número de turno y la información sobre el tiempo estimado de espera.

- **Presencial:** El cliente recibe el ticket impreso y espera en la sala de espera.
- **En Línea:** El cliente recibe una notificación digital con el número de turno y el tiempo estimado.

Espera de Atención:

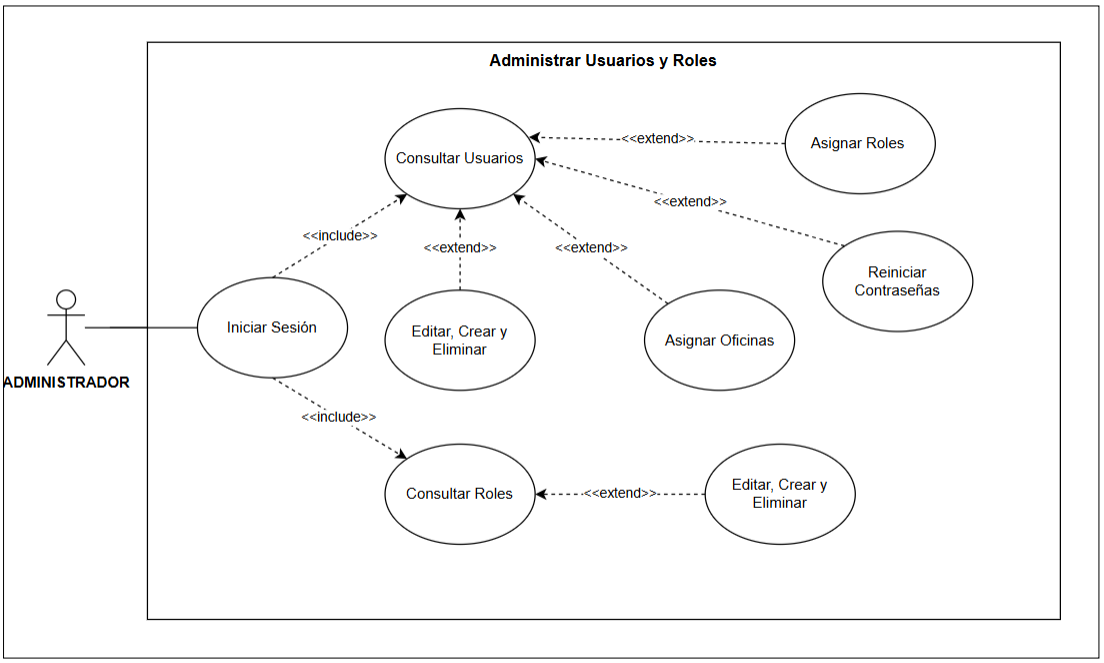
- El cliente espera a ser llamado por el sistema.

Flujos Alternativos:

Calificación de la Atención: Una vez que el turno ha sido atendido, el cliente puede calificar la atención recibida.

- **Presencial:** El cliente puede calificar la atención mediante una encuesta digital en la oficina.
- **En Línea:** El cliente puede calificar la atención a través de un enlace enviado por correo electrónico.

Caso de uso Administrar Usuarios y Roles



Caso de Uso: Administrar Usuarios y Roles

Descripción: Este caso de uso detalla cómo un administrador gestiona usuarios y roles.

Actores: Administrador, Sistema de Gestión.

Precondiciones: El administrador debe loguearse en el sistema.

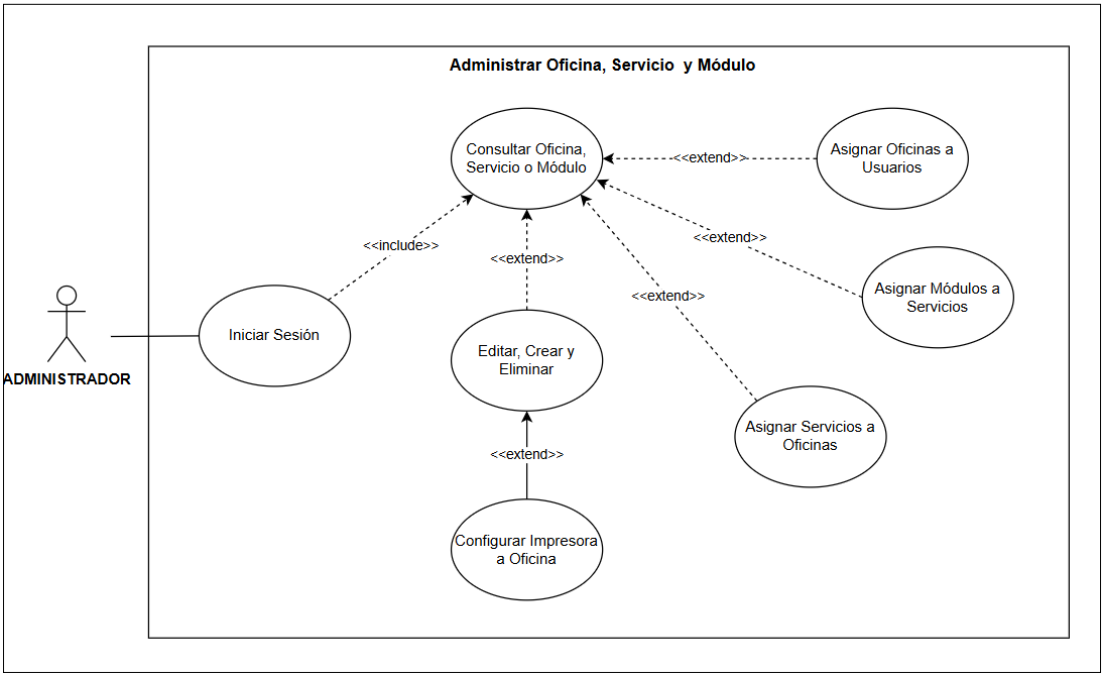
Flujo Principal:

- El administrador consulta la lista de usuarios.
- El administrador asigna roles a los usuarios.
- El administrador puede reiniciar contraseñas si es necesario.
- El administrador puede asignar oficinas a los usuarios.

Flujos Alternativos:

Si el administrador necesita crear, editar o eliminar usuarios o roles, puede hacerlo desde la interfaz de administración.

Caso de uso Administrar Oficina, Servicio y Módulo



Caso de Uso: Administrar Oficina, Servicio y Módulo

Descripción: Este caso de uso detalla el proceso de administración de oficinas, servicios y módulos por parte de un administrador.

Actores: Administrador, Sistema de Gestión.

Precondiciones: El administrador debe loguearse en el sistema.

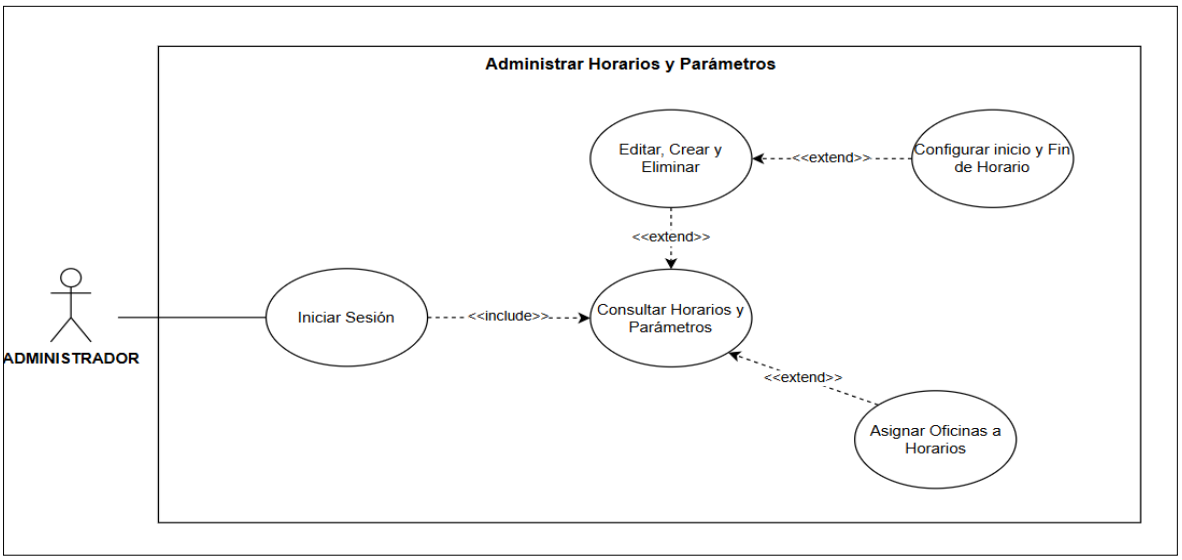
Flujo Principal:

- El administrador consulta la lista de oficinas, servicios y módulos.
- El administrador asigna oficinas a usuarios.
- El administrador asigna módulos a servicios.
- El administrador asigna servicios a oficinas.
- El administrador puede editar, crear o eliminar oficinas, servicios o módulos.

Flujos Alternativos:

Si el administrador necesita configurar una impresora para una oficina, puede hacerlo desde la interfaz de administración.

Caso de uso Administrar Horarios y Parámetros



Caso de Uso: Administrar Horarios y Parámetros

Descripción: Este caso de uso detalla la forma de administración de horarios y parámetros del sistema por parte de un administrador.

Actores: Administrador, Sistema de Gestión.

Precondiciones: El administrador debe loguearse en el sistema.

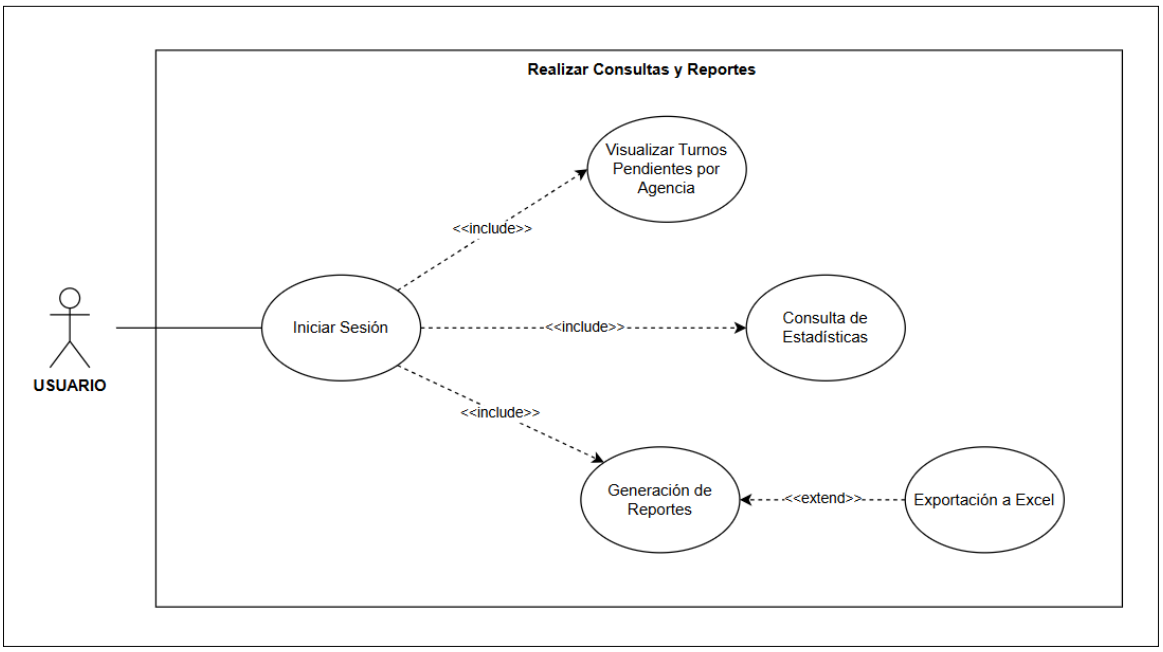
Flujo Principal:

- El administrador consulta los horarios y parámetros actuales.
- El administrador asigna horarios a oficinas.
- El administrador puede editar, crear o eliminar horarios y parámetros.

Flujos Alternativos:

Si el administrador necesita configurar parámetros específicos, puede hacerlo desde la interfaz de administración.

Caso de uso Realizar Consultas y Reportes



Caso de Uso: Realizar Consultas y Reportes

Descripción: Este caso de uso se detalla la forma de generación de consultas y reportes por parte de un administrador o usuario autorizado.

Actores: Administrador, Usuario, Sistema de Gestión.

Flujo Principal:

- El usuario visualiza los turnos pendientes por agencia.
- El usuario consulta estadísticas de atención.
- El usuario genera reportes.
- El usuario exporta los reportes a Excel.

Flujos Alternativos:

Si el usuario necesita realizar consultas específicas, puede filtrar los datos según sus necesidades.

2.4. Restricciones

- Tecnológicas: Uso obligatorio del stack MEAN.
- Seguridad: Cifrado de contraseñas con bcrypt y autenticación JWT.
- Hardware: Compatibilidad con dispositivos móviles y tablets.

2.5. Suposiciones y dependencias

- Se asume que las cooperativas cuentan con conexión a Internet estable.
- Depende de MongoDB para el almacenamiento de datos y de Node.js para el backend.

3. Requisitos específicos

3.1. Requerimientos Funcionales

Identificación del requerimiento:	**RF01**
Nombre del Requerimiento:	Autentificación de Usuario.
Características:	Los usuarios deberán identificarse para acceder a cualquier parte del sistema.
Descripción del requerimiento:	El sistema solicitará un nombre de usuario y contraseña. La contraseña debe estar enmascarada con opción de visualización.
Requerimiento NO funcional:	• RNF01 • RNF02 • RNF05 • RNF08
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	**RF02**
Nombre del Requerimiento:	Cambio de Contraseña.
Características:	El sistema debe permitir al usuario cambiar su contraseña y evaluar su seguridad.
Descripción del requerimiento:	Validación de seguridad de la contraseña (débil, segura, muy segura).
Requerimiento NO funcional:	• RNF04 • RNF05

Prioridad del requerimiento:	Alta

Identificación del requerimiento:	**RF03**
Nombre del Requerimiento:	Pantalla de Bienvenida.
Características:	Mostrar nombre, oficina del usuario y logo institucional.
Descripción del requerimiento:	La pantalla de inicio debe mostrar el nombre del usuario y su oficina. Debe incluir un menú y el logo en el centro.
Requerimiento NO funcional:	-
Prioridad del requerimiento:	Media

Identificación del requerimiento:	**RF04**
Nombre del Requerimiento:	Asignación de Turnos Presenciales y en Línea.
Características:	Asignación secuencial con código de tres letras.
Descripción del requerimiento:	El sistema debe permitir tomar turnos en agencias y vía web.
Requerimiento NO funcional:	-
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	**RF05**
Nombre del Requerimiento:	Visualización de Turnos Pendientes.
Características:	Mostrar turnos en espera o mensaje si no hay asignaciones.
Descripción del requerimiento:	Se debe indicar que no hay turnos si no existen asignaciones en el día.
Requerimiento NO funcional:	-

Prioridad del requerimiento:	Media

Identificación del requerimiento:	**RF06**
Nombre del Requerimiento:	Transferencia de Turnos.
Características:	Transferencia entre módulos en tiempo real.
Descripción del requerimiento:	Un turno llamado puede ser transferido a otro módulo de la misma agencia.
Requerimiento NO funcional:	-
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	**RF07**
Nombre del Requerimiento:	Solicitud de Turno en Línea con Validación.
Características:	Validación de identidad, envío de OTP y verificación de horario.
Descripción del requerimiento:	El sistema debe solicitar identificación, número telefónico y correo. Enviar OTP.
Requerimiento NO funcional:	• RNF07
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	**RF08**
Nombre del Requerimiento:	Visualización de Turnos en Módulo.
Características:	Actualización en tiempo real y resaltado de turnos en espera.
Descripción del requerimiento:	Lista de turnos actualizada y resaltada en tiempo real.
Requerimiento NO funcional:	-
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RF09
Nombre del Requerimiento:	Gestión de Oficinas y Módulos.
Características:	Gestión de elementos físicos y configuraciones básicas.
Descripción del requerimiento:	Creación, edición y eliminación de oficinas, servicios y módulos. Configuración de voz e IP.
Requerimiento NO funcional:	• RNF16
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RF10
Nombre del Requerimiento:	Gestión de Usuarios y Roles.
Características:	Asignación de roles y permisos a usuarios.
Descripción del requerimiento:	Creación, edición y eliminación de usuarios y roles.
Requerimiento NO funcional:	• RNF05
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RF11
Nombre del Requerimiento:	Configuración de Horarios por Oficina.
Características:	Horarios ajustables de lunes a domingo.
Descripción del requerimiento:	Asignación flexible de horarios a oficinas.
Requerimiento NO funcional:	-
Prioridad del requerimiento:	Media

Identificación del requerimiento:	RF12
Nombre del Requerimiento:	Actualización de Lista de Turnos.
Características:	Notificaciones audiovisuales y actualización automática.
Descripción del requerimiento:	Lista debe actualizarse automáticamente. Mostrar notificaciones.
Requerimiento NO funcional:	• RNF14
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RF13
Nombre del Requerimiento:	Reportes de Gestión de Turnos.
Características:	Estadísticas exportables en PDF y Excel.
Descripción del requerimiento:	Reportes sobre turnos atendidos, tiempos de espera y eficiencia operativa.
Requerimiento NO funcional:	-
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RF14
Nombre del Requerimiento:	Actualización en Tiempo Real de Turnos.
Características:	Notificaciones y actualizaciones automáticas.
Descripción del requerimiento:	Lista debe actualizarse y emitir notificaciones audiovisuales.
Requerimiento NO funcional:	• RNF14
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	**RF15**
Nombre del Requerimiento:	Visualización de Tiempo de Atención.
Características:	Cálculo automático del tiempo de atención.
Descripción del requerimiento:	Mostrar tiempo desde que se llama hasta que se atiende el turno.
Requerimiento NO funcional:	-
Prioridad del requerimiento:	Media

Identificación del requerimiento:	**RF16**
Nombre del Requerimiento:	Validación de Servicio Activo en Operador.
Características:	Prevención de múltiples servicios simultáneos.
Descripción del requerimiento:	El sistema debe impedir que un operador inicie otro servicio si ya tiene uno activo.
Requerimiento NO funcional:	-
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	**RF17**
Nombre del Requerimiento:	Registro de Tiempos de Turno.
Características:	Registro para reportes y estadísticas.
Descripción del requerimiento:	Registrar tiempos desde toma, llamado y atención de turno.
Requerimiento NO funcional:	• RNF17
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RF18
Nombre del Requerimiento:	Código QR en Turnos Impresos.
Características:	Código QR escaneable con información del turno.
Descripción del requerimiento:	Incluir código QR con código, servicio y fecha/hora.
Requerimiento NO funcional:	-
Prioridad del requerimiento:	Media

Identificación del requerimiento:	RF19
Nombre del Requerimiento:	Validación de Horario en Turnos en Línea.
Características:	Validación de horario según oficina.
Descripción del requerimiento:	Validar que el turno se tome dentro del horario de atención.
Requerimiento NO funcional:	-
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RF20
Nombre del Requerimiento:	CAPTCHA ante Actividades Sospechosas.
Características:	CAPTCHA como medida de seguridad.
Descripción del requerimiento:	Mostrar CAPTCHA al detectar acciones inusuales.
Requerimiento NO funcional:	• RNF04
Prioridad del requerimiento:	Media

Identificación del requerimiento:	RF21
Nombre del Requerimiento:	Confirmación por Correo de Turno.
Características:	Correo automático tras toma de turno.
Descripción del requerimiento:	Enviar información del turno al correo del cliente.
Requerimiento NO funcional:	-
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RF22
Nombre del Requerimiento:	Visualización de Turnos en Módulo.
Características:	Lista actualizada en tiempo real.
Descripción del requerimiento:	Actualizar en tiempo real y resaltar turnos nuevos.
Requerimiento NO funcional:	-
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RF23
Nombre del Requerimiento:	Configuración de Voz por Oficina.
Características:	Selección de voz masculina o femenina.
Descripción del requerimiento:	Permitir configurar voz según oficina.
Requerimiento NO funcional:	-
Prioridad del requerimiento:	Media

Identificación del requerimiento:	**RF24**
Nombre del Requerimiento:	Validación de Identificación Presencial.
Características:	Solicitar ID si la oficina lo requiere.
Descripción del requerimiento:	Validar identificación en oficinas que lo requieran.
Requerimiento NO funcional:	-
Prioridad del requerimiento:	Media

Identificación del requerimiento:	**RF25**
Nombre del Requerimiento:	Reinicio de Contraseña con Clave Temporal.
Características:	Clave temporal enviada al correo.
Descripción del requerimiento:	Enviar clave temporal y forzar cambio al inicio.
Requerimiento NO funcional:	• RNF04 • RNF07
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	**RF26**
Nombre del Requerimiento:	Configuración de Horarios Diarios.
Características:	Configuración individual por día.
Descripción del requerimiento:	Asignar horarios diferentes a cada día de la semana.
Requerimiento NO funcional:	-
Prioridad del requerimiento:	Media

3.2. Requerimientos No Funcionales

Identificación del requerimiento:	**RNF01**
Nombre del Requerimiento:	Rendimiento y Escalabilidad
Características:	Base de datos no relacional optimizada para múltiples usuarios simultáneos.
Descripción del requerimiento:	El sistema debe soportar hasta 300 usuarios concurrentes, responder consultas en menos de 200 ms y soportar 100,000 consultas diarias.
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	**RNF02**
Nombre del Requerimiento:	Escalabilidad Vertical
Características:	Permitir escalabilidad sin afectar funcionamiento.
Descripción del requerimiento:	La aplicación debe ser capaz de escalar verticalmente según la demanda.
Prioridad del requerimiento:	Media

Identificación del requerimiento:	**RNF03**
Nombre del Requerimiento:	Integración con Otros Sistemas
Características:	Facilidad de integración gracias a APIs bien documentadas.
Descripción del requerimiento:	Contar con APIs documentadas para integración con otros sistemas.
Prioridad del requerimiento:	Media

Identificación del requerimiento:	RNF04
Nombre del Requerimiento:	Almacenamiento Seguro de Contraseñas
Características:	Cifrado de contraseñas con método seguro como bcrypt.
Descripción del requerimiento:	Las contraseñas deben cifrarse con bcrypt u otro método seguro.
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RNF05
Nombre del Requerimiento:	Controles de Acceso Basados en Roles
Características:	RBAC para asignar permisos según el rol.
Descripción del requerimiento:	Permitir asignación de roles y permisos a usuarios.
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RNF06
Nombre del Requerimiento:	Comunicación Segura por HTTPS
Características:	Uso obligatorio de HTTPS.
Descripción del requerimiento:	El sistema debe usar HTTPS en todas las transacciones.
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RNF07
Nombre del Requerimiento:	Autenticación con JWT
Características:	Uso de JSON Web Tokens para autenticación segura.
Descripción del requerimiento:	Garantizar la integridad y confidencialidad de credenciales con JWT.
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RNF08
Nombre del Requerimiento:	Alta Disponibilidad del Sistema
Características:	99.9% de disponibilidad en horario laboral.
Descripción del requerimiento:	Disponibilidad de 08:00 a 18:00 con máximo 0.1% de inactividad.
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RNF09
Nombre del Requerimiento:	Recuperación ante Fallos
Características:	Backups diarios del sistema.
Descripción del requerimiento:	Copias de seguridad diarias de la base de datos y aplicación.
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RNF10
Nombre del Requerimiento:	Compatibilidad con Dispositivos Móviles
Características:	Diseño responsivo compatible con PrimeFlex.
Descripción del requerimiento:	La interfaz debe adaptarse a móviles y tabletas.
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RNF11
Nombre del Requerimiento:	Generación de Turno en Menos de 3 Clics
Características:	Usabilidad eficiente sin necesidad de capacitación.
Descripción del requerimiento:	Un usuario debe generar un turno en menos de 3 clics.
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RNF12
Nombre del Requerimiento:	Mensajes de Ayuda y Validación en Tiempo Real
Características:	Validaciones y ayudas visuales inmediatas.
Descripción del requerimiento:	Mensajes y validaciones deben mostrarse en tiempo real.
Prioridad del requerimiento:	Media

Identificación del requerimiento:	RNF13
Nombre del Requerimiento:	Carga de Páginas Rápida
Características:	Carga en menos de 3 segundos.
Descripción del requerimiento:	Las páginas deben cargarse en menos de 3 segundos en conexiones estándar.
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RNF14
Nombre del Requerimiento:	Indicadores Visuales para Procesos Lentos
Características:	Spinners o barras de progreso visibles.
Descripción del requerimiento:	Mostrar indicadores visuales para acciones >1 segundo.
Prioridad del requerimiento:	Media

Identificación del requerimiento:	RNF15
Nombre del Requerimiento:	Historial de Actualizaciones
Características:	Acceso al historial de cambios del sistema.
Descripción del requerimiento:	Visualización de cambios realizados y sus fechas.
Prioridad del requerimiento:	Media

Identificación del requerimiento:	RNF16
Nombre del Requerimiento:	Arquitectura Modular
Características:	Permite actualización independiente de componentes.
Descripción del requerimiento:	Actualización de componentes sin afectar el sistema.
Prioridad del requerimiento:	Alta

Identificación del requerimiento:	RNF17
Nombre del Requerimiento:	Logs y Monitoreo de Fallos
Características:	Generación de logs y monitoreo en tiempo real.
Descripción del requerimiento:	Detectar fallos mediante registros y monitoreo continuo.
Prioridad del requerimiento:	Alta

3.3. Requisitos comunes de las interfaces

3.1.1 Interfaces de usuario

Las interfaces con las que interactuará el usuario consistirán en un grupo de pantallas con componentes como botones, tablas y campos de texto. Estas serán realizadas específicamente para el presente sistema y podrán ser visualizadas desde un navegador dentro de la red interna o desde internet.

3.1.2 Interfaces de hardware

Será indispensable para el acceso disponer de dispositivos en un perfecto estado con acceso a la red, con un navegador actualizado, con mínimo 2GB de memoria RAM, y métodos de entrada adecuados como mouse y teclado o una pantalla táctil dependiendo del dispositivo en uso.

3.1.3 Interfaces de software

Compatible con sistemas operativos modernos como Windows 10 o superior, Linux, macOS, Android, o iOS. Para dispositivos móviles, el sistema está optimizado para navegadores móviles modernos.

Navegadores compatibles

Navegadores actualizados que soporten HTML5, JavaScript y CSS3:

- Google Chrome (v90 o superior)
- Mozilla Firefox (v80 o superior)
- Microsoft Edge (Chromium)
- Safari (en iOS/macOS)

3.1.4 Interfaces de comunicación

La comunicación entre los diferentes componentes del sistema se realizará mediante protocolos estándar de internet. El navegador web se comunicará con el servidor backend a través del protocolo HTTP/HTTPS, utilizando peticiones RESTful con datos en formato JSON. La interacción entre el servidor y la base de datos MongoDB se llevará a cabo mediante el uso de drivers nativos sobre conexión TCP/IP.

ANEXO B. Documentación Técnica API (Backend)

A continuación, describe la arquitectura y los principales endpoints del backend desarrollado en Node.js. Se presenta un resumen estructurado en tablas para cada módulo que forma parte del backend, para mayor detalle de codificación se puede visitar la siguiente url del repositorio publicado en github donde se encuentra todo el código fuente: *https://github.com/navidaro/SistemaTurnos*

Resumen de Endpoints más importantes

Cada tabla resume los endpoints más relevantes del módulo.

Módulo: Autenticación

Gestión de autenticación y usuarios (registro, login, cambio de contraseña).

Método	Ruta	Descripción
POST	/register	Registra un nuevo usuario
POST	/login	Procesa login de usuario
POST	/register-admin	Registra un nuevo usuario
POST	/update-password/:id	Crea o procesa recurso en auth

Módulo: Modulos

Administración de módulos asociados a servicios.

Método	Ruta	Descripción
GET	/	Obtiene recurso en modulo
GET	/servicio/:servicioId	Obtiene recurso en modulo
GET	/:id	Obtiene recurso en modulo
POST	/	Crea o procesa recurso en modulo
PUT	/:id	Actualiza recurso en modulo
DELETE	/:id	Elimina recurso en modulo

Módulo: Roles

Administración de roles y permisos.

Método	Ruta	Descripción
GET	/	Obtiene recurso en roles
GET	/:id	Obtiene recurso en roles
POST	/	Crea o procesa recurso en roles
PUT	/:id	Actualiza recurso en roles
DELETE	/:id	Elimina recurso en roles

Módulo: Turno

Gestión y control de turnos presenciales.

Método	Ruta	Descripción
GET	/turno/crear	Crea un nuevo recurso o turno
POST	/crear	Crea un nuevo recurso o turno
POST	/llamarturno	Llama al siguiente turno
PUT	/turnosestado/:oficinaId	Actualiza recurso en turno
PUT	/numeroturnos	Actualiza recurso en turno
POST	/actualizarestadoturno	Crea o procesa recurso en turno
POST	/transferir	Crea o procesa recurso en turno

Módulo: Turno en Línea

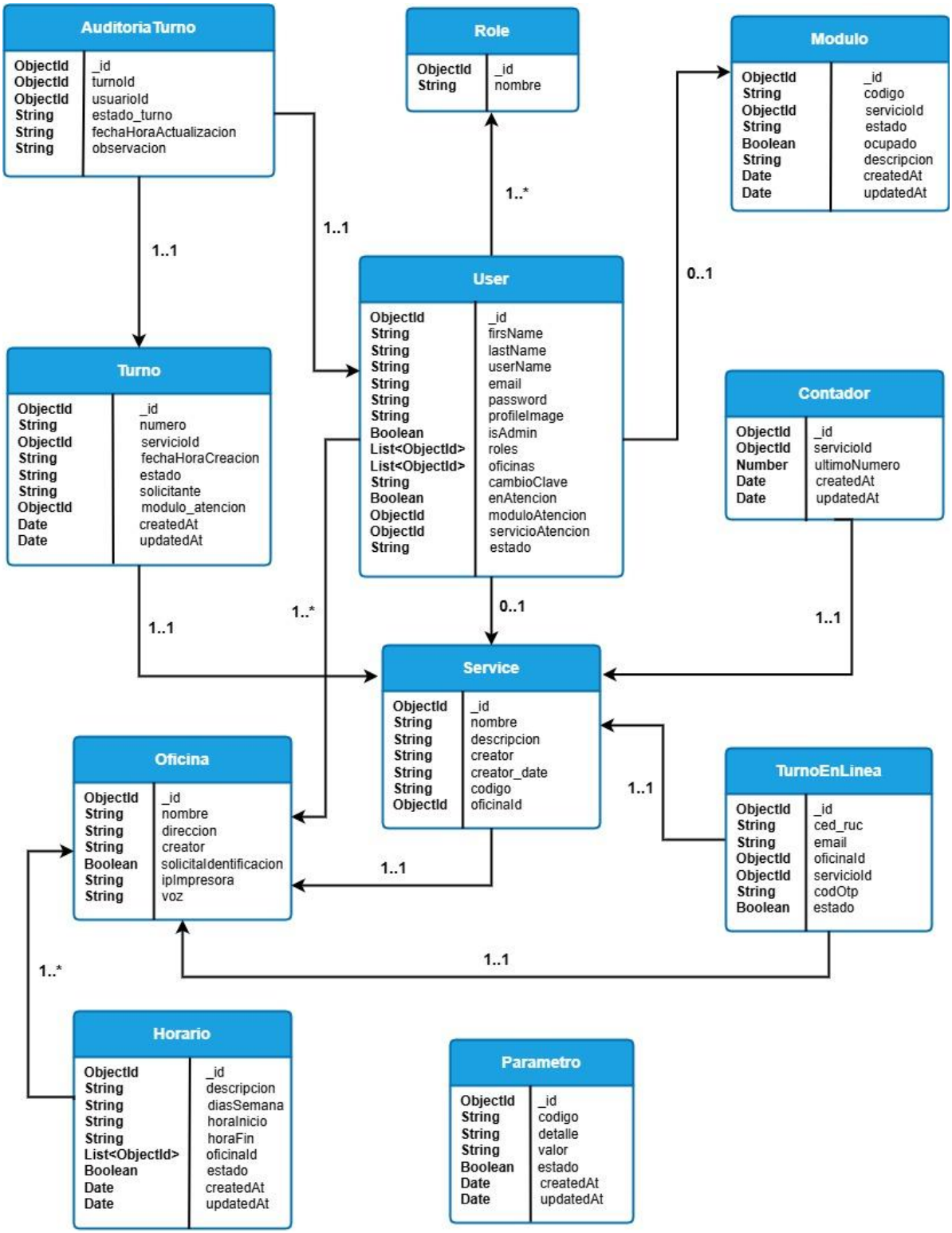
Gestión de turnos en línea y validación OTP.

Método	Ruta	Descripción
GET	/	Obtiene recurso en turnoEnLinea
POST	/	Crea o procesa recurso en turnoEnLinea
POST	/validarotp	Valida OTP para turnos en línea
POST	/:id	Crea o procesa recurso en turnoEnLinea
GET	/oficinas	Obtiene recurso en turnoEnLinea

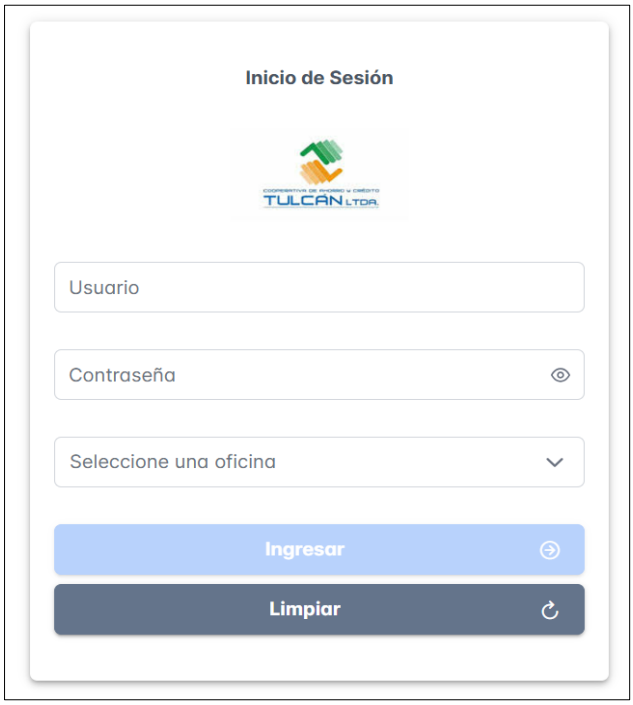
ANEXO C. Desarrollo del sistema de gestión de turnos

C 1. Sprint 1: Fundamentos del Sistema

RNF01 - Diagrama de la base de datos

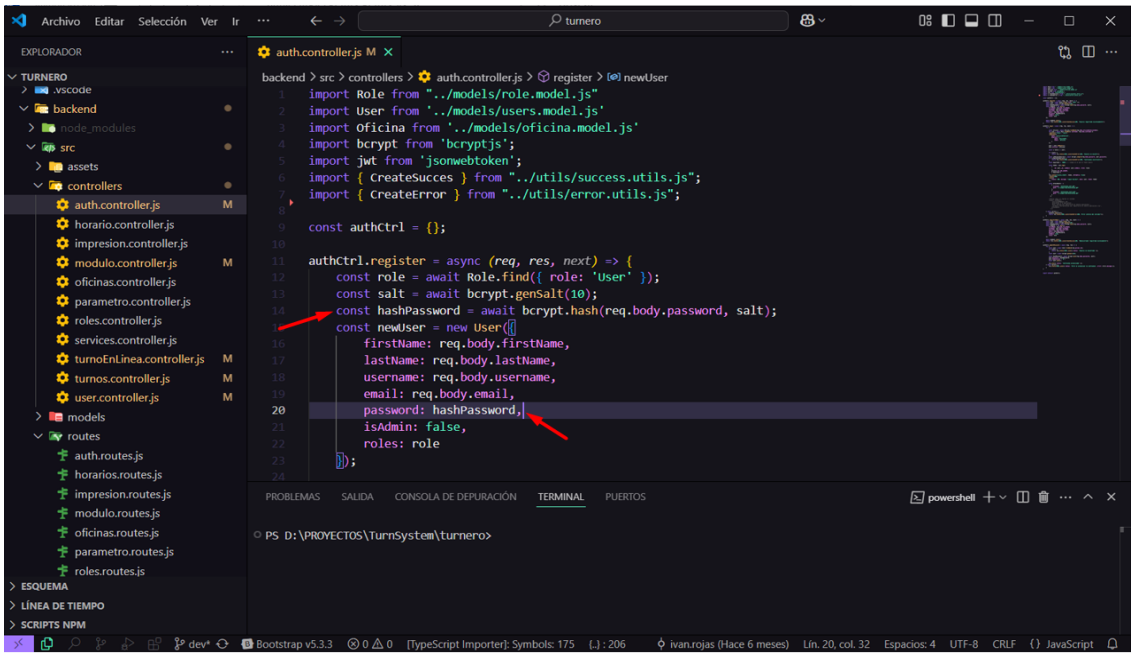


RF01 - Pantalla de inicio de sesión

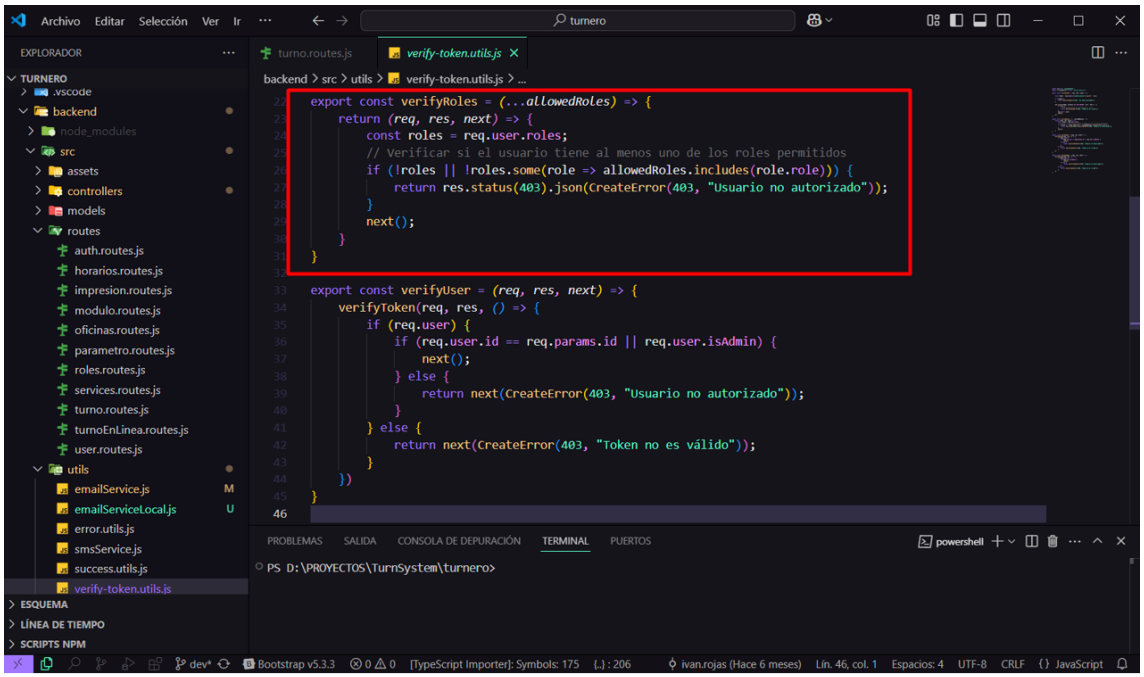


RF03 - Pantalla de bienvenida al sistema

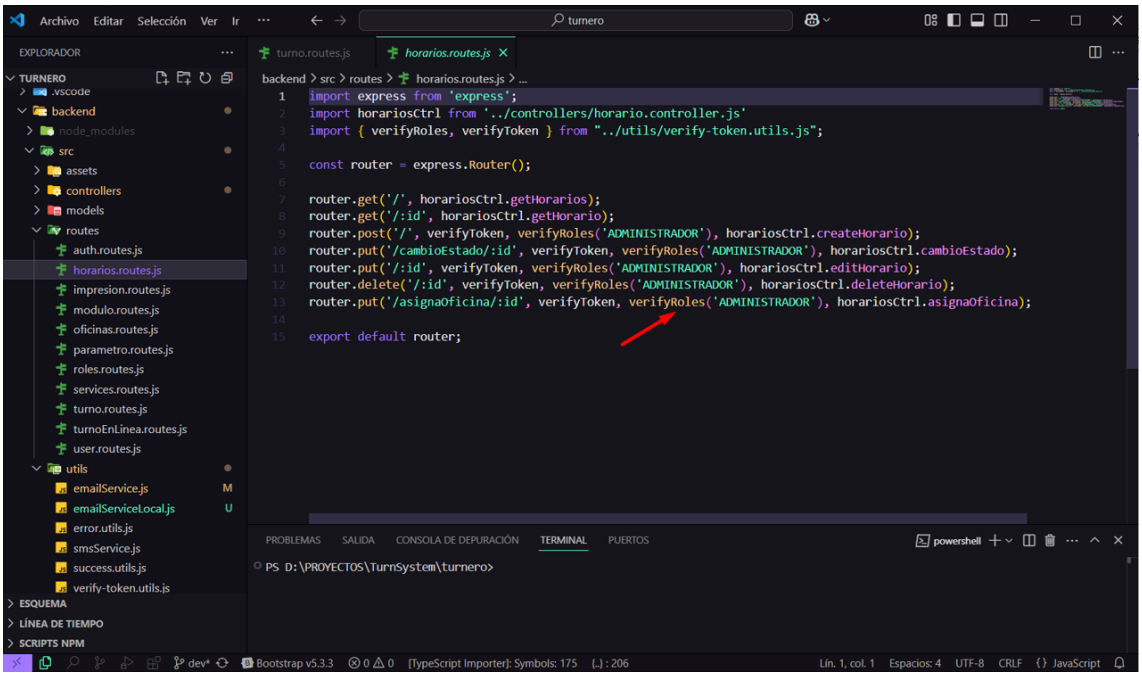
RNF04 - Cifrado de contraseñas usando Bcrypt



RNF05 - Control de acceso basado en roles (RBAC)

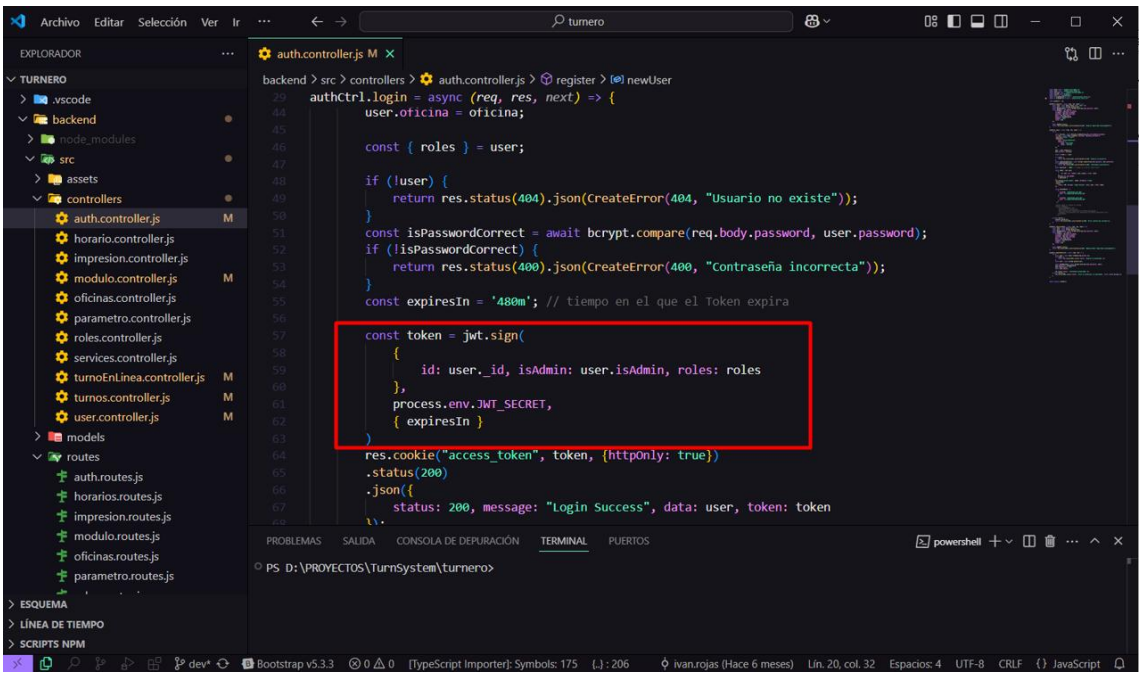


RNF05 - Middleware de verificación de roles asignados

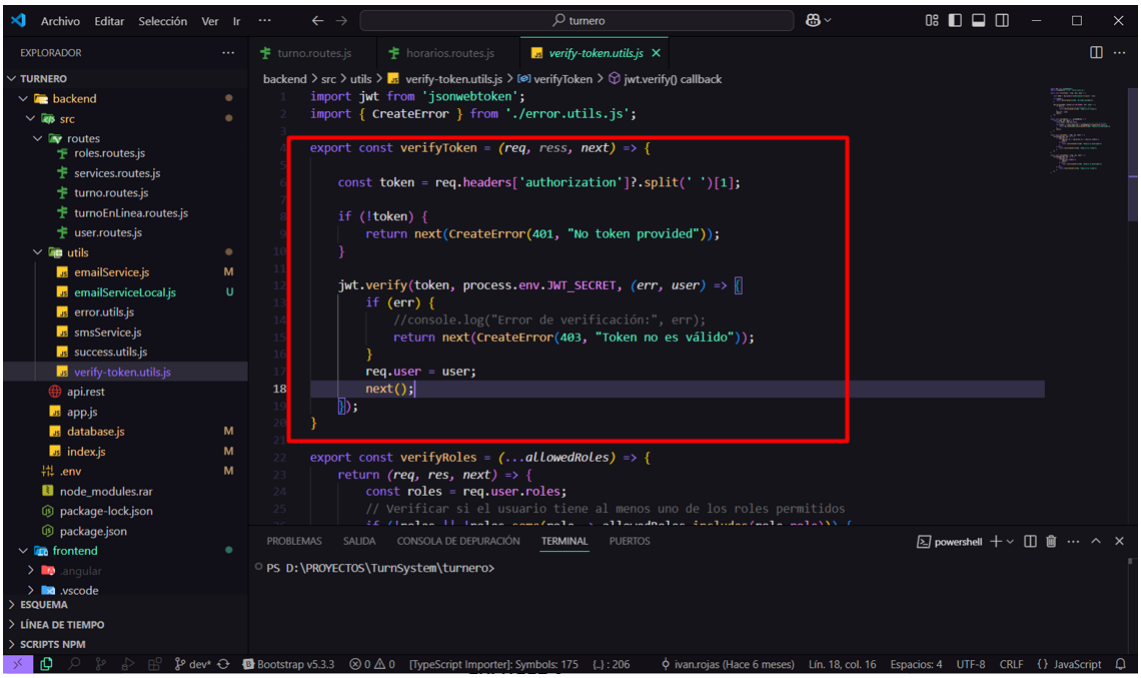


RNF07 - Autenticación por medio de JSON Web Token (JWT)

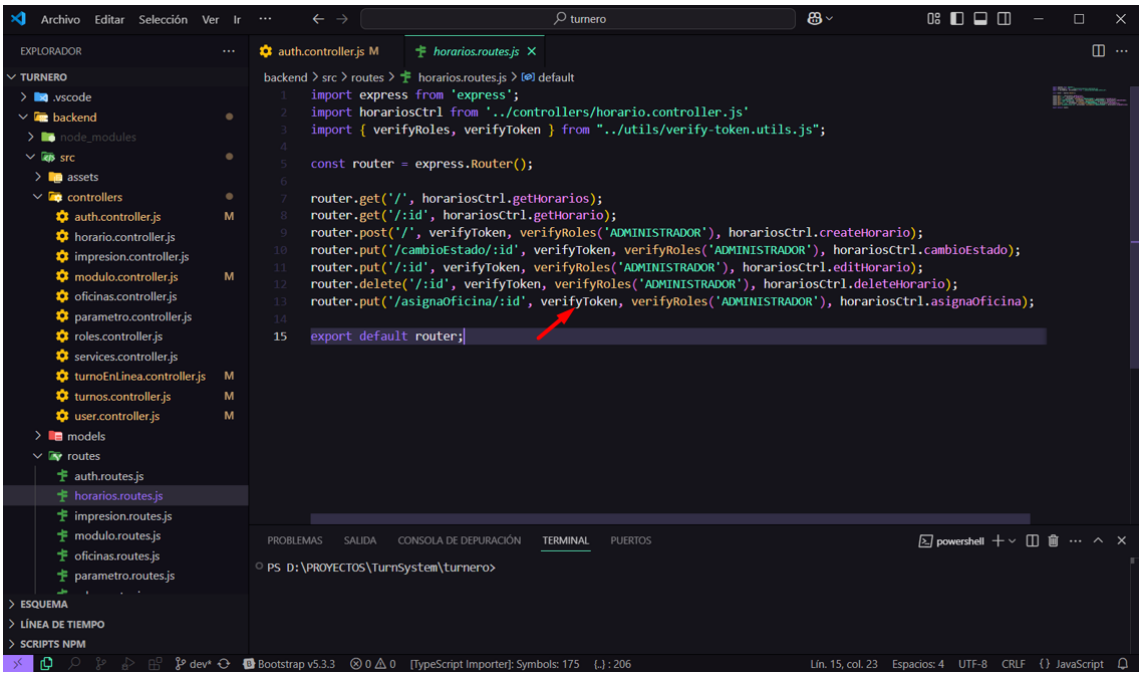
Se provee del token al usuario al iniciar sesión



Middleware para verificar validez del token recibido

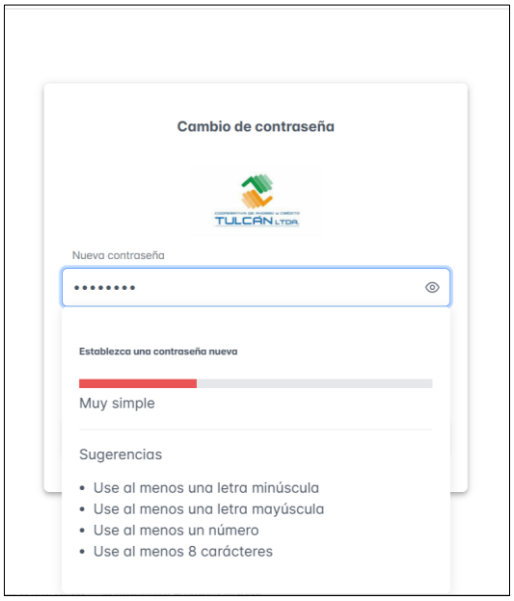


En cada ruta se invoca el middleware para verificar el token

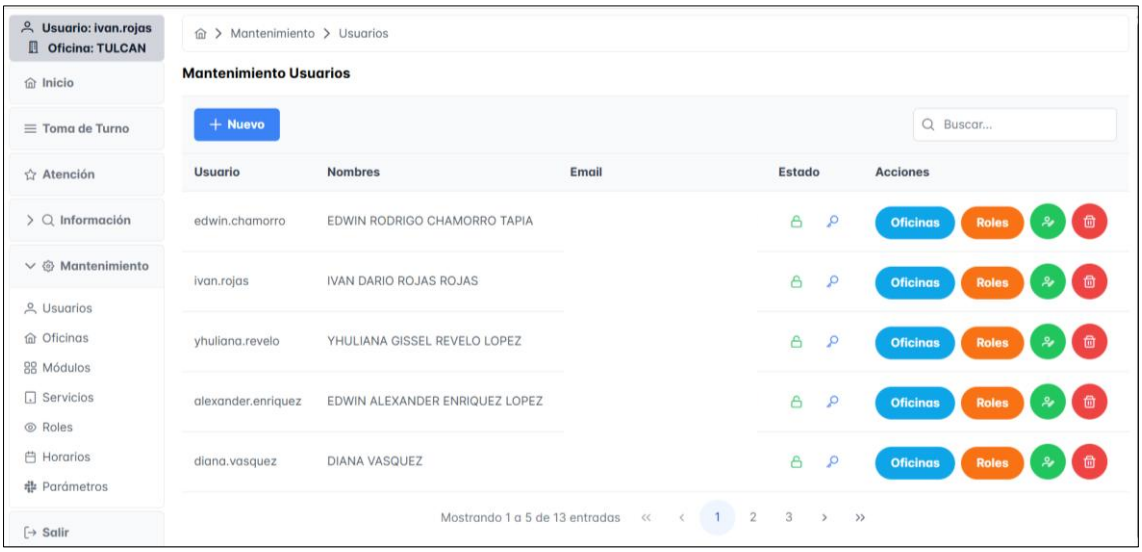


C 2. Sprint 2: Gestión de Usuarios

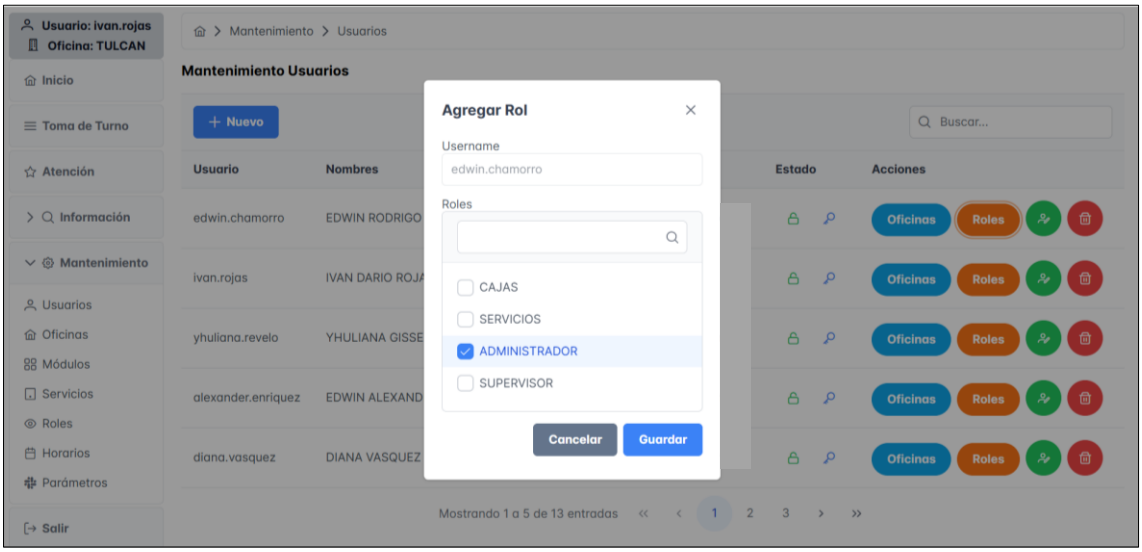
RF02 - Cambio de contraseña en el sistema



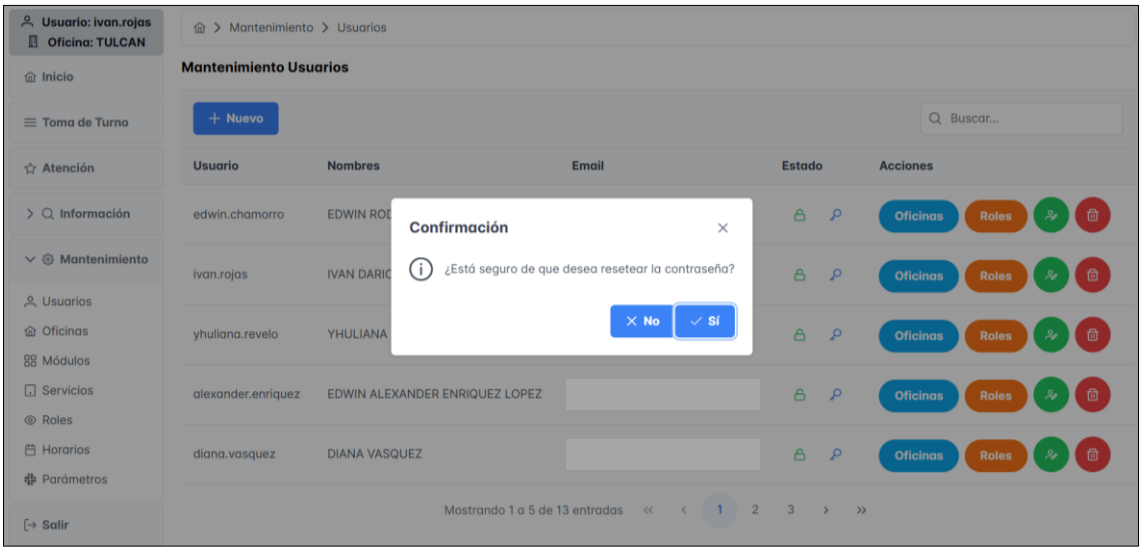
RF10 – Administración de usuarios (CRUD)



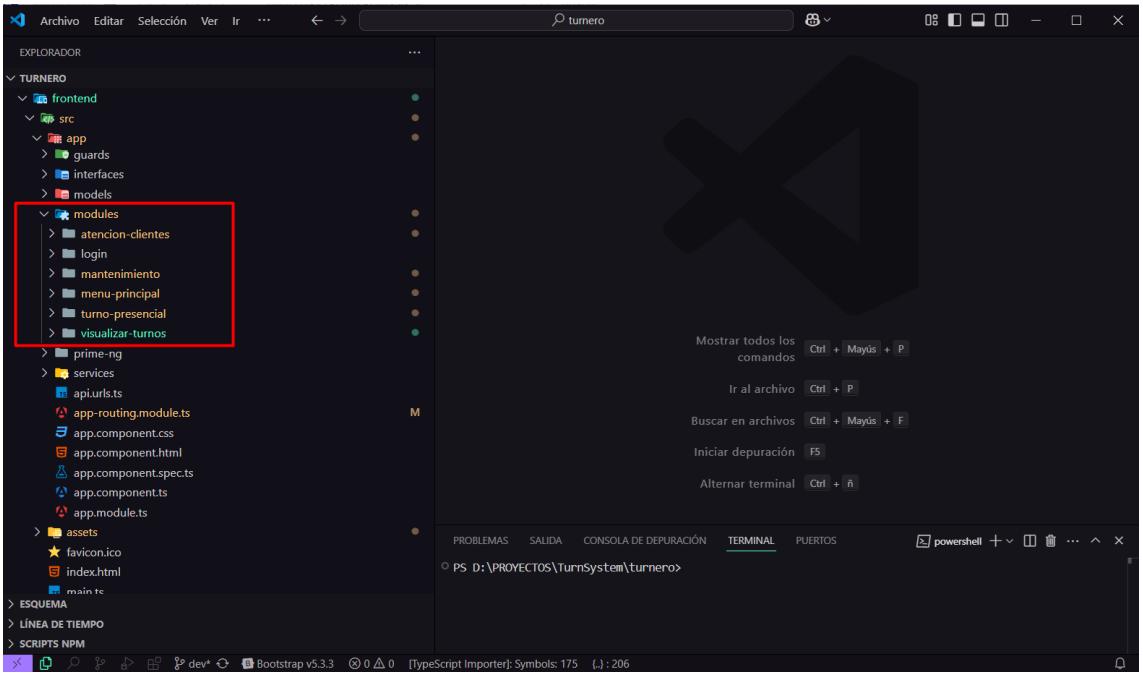
RF10 – Administración de roles



RF25 - Reinicio de contraseña



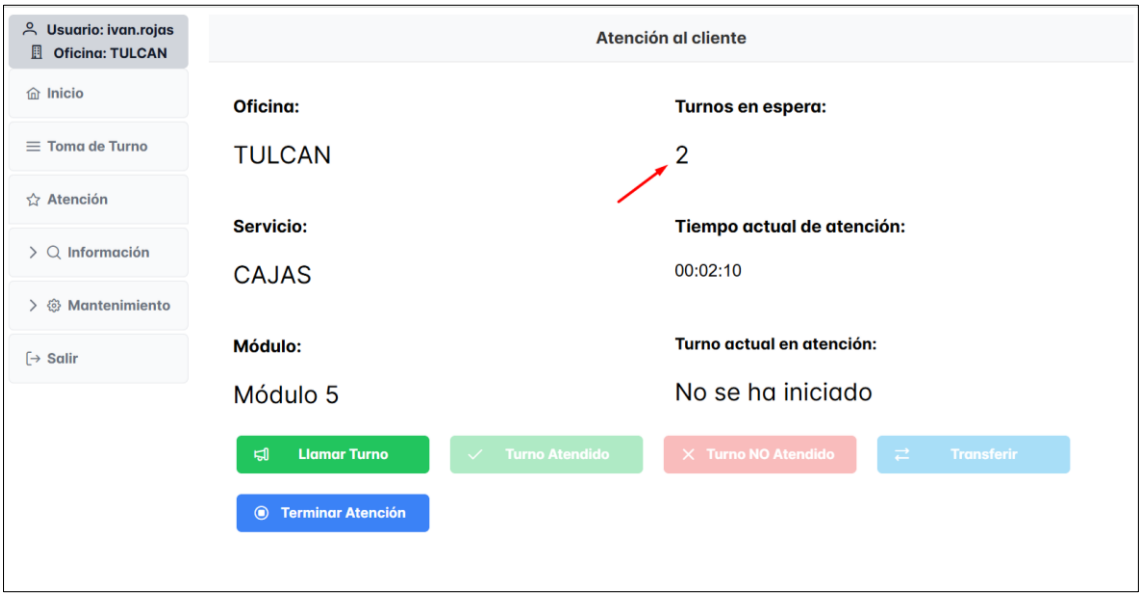
RNF16 - Arquitectura de aplicación modular



C 3. Sprint 3: Turnos Básicos

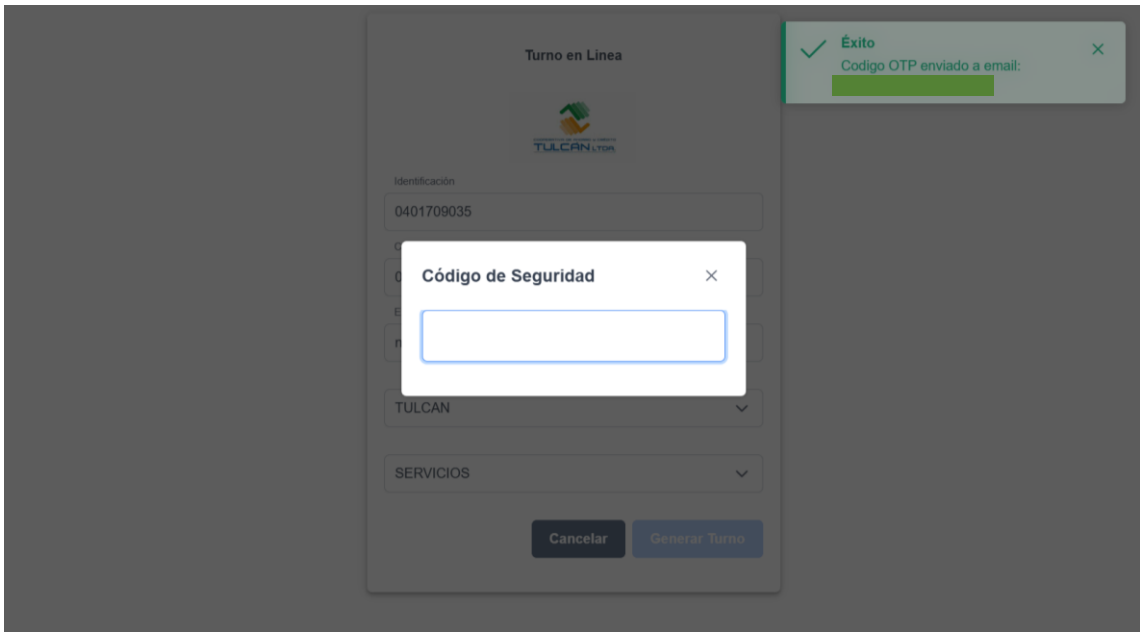
RF04 - Toma de turno presencial

RF05 - Visualización de turnos en espera

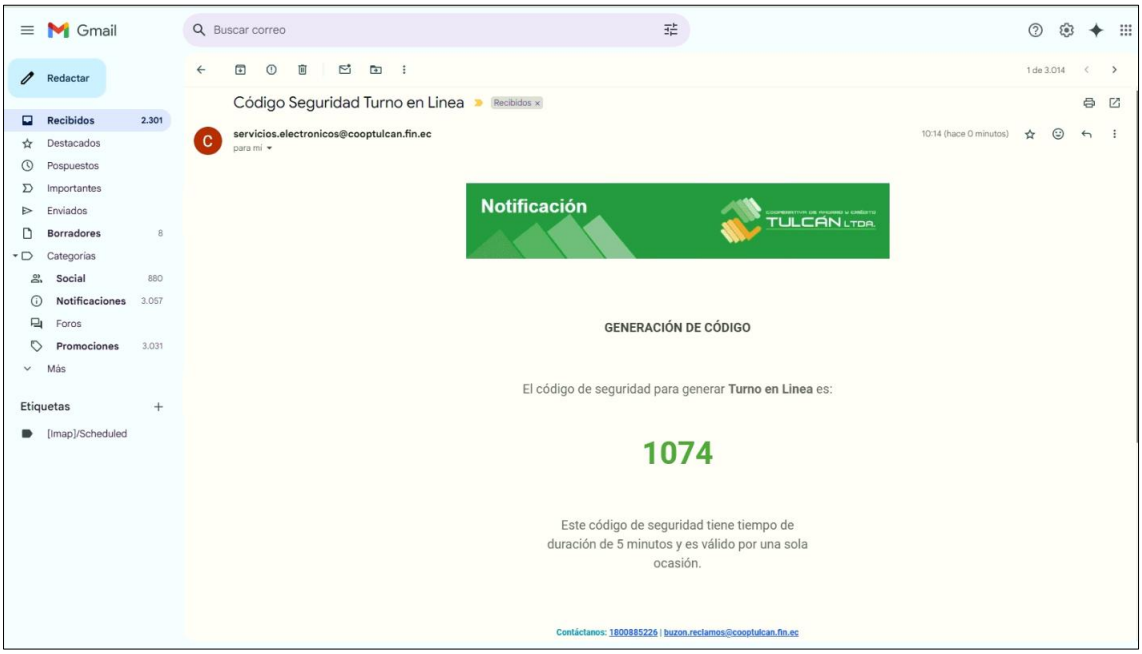


C 4. Sprint 4: Turnos en Línea

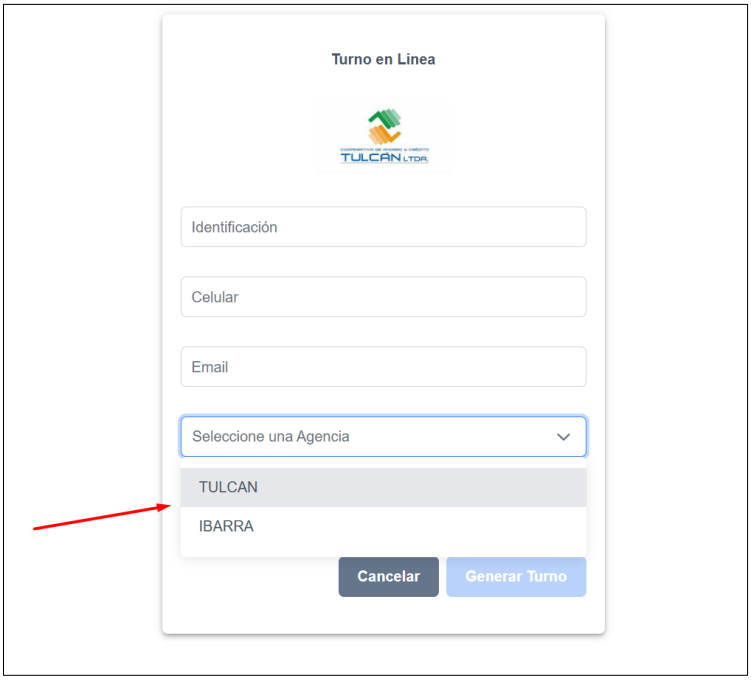
RF08 - Envío de Contraseña de un Solo Uso (OTP)



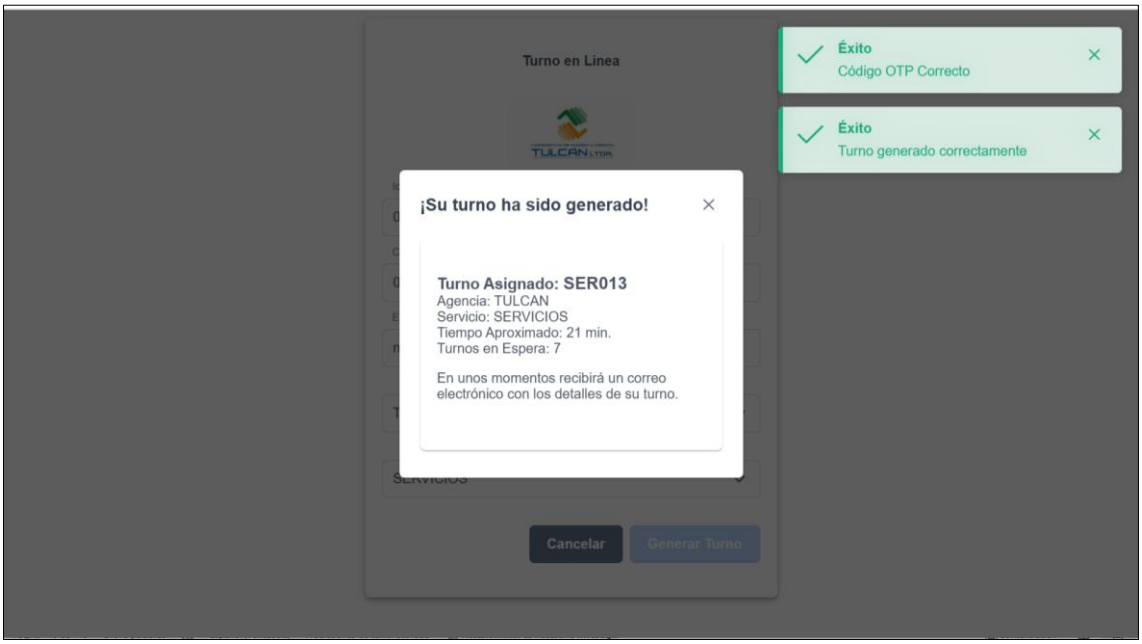
RF08 - Recepción de código OTP al correo registrado



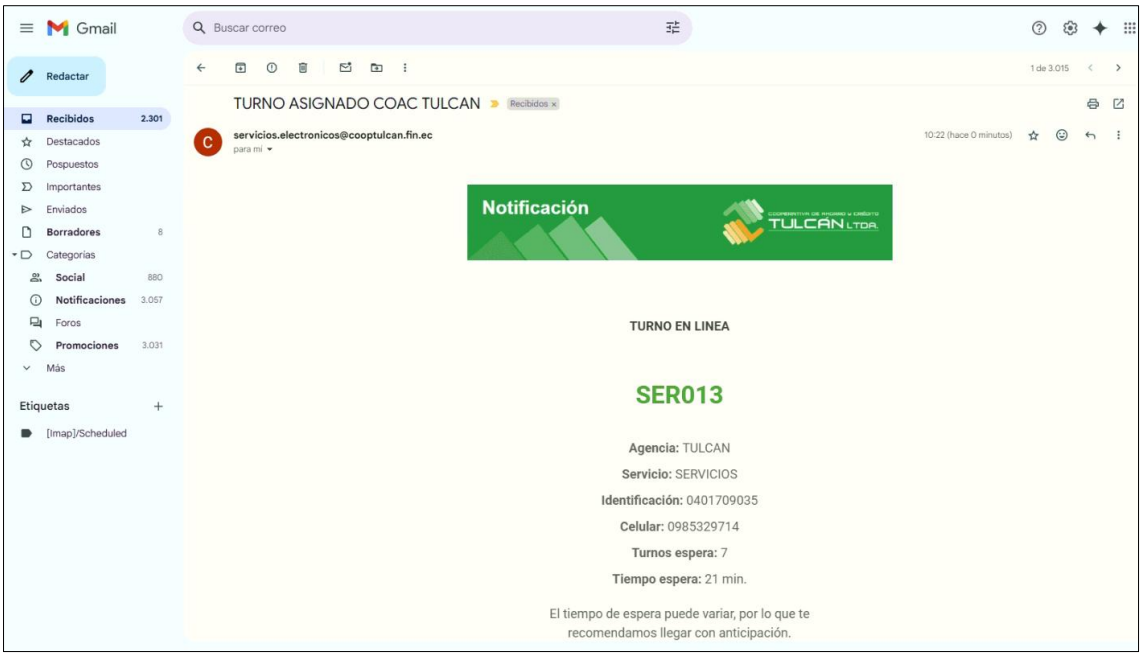
RF19 - Validación de horarios de oficinas en atención



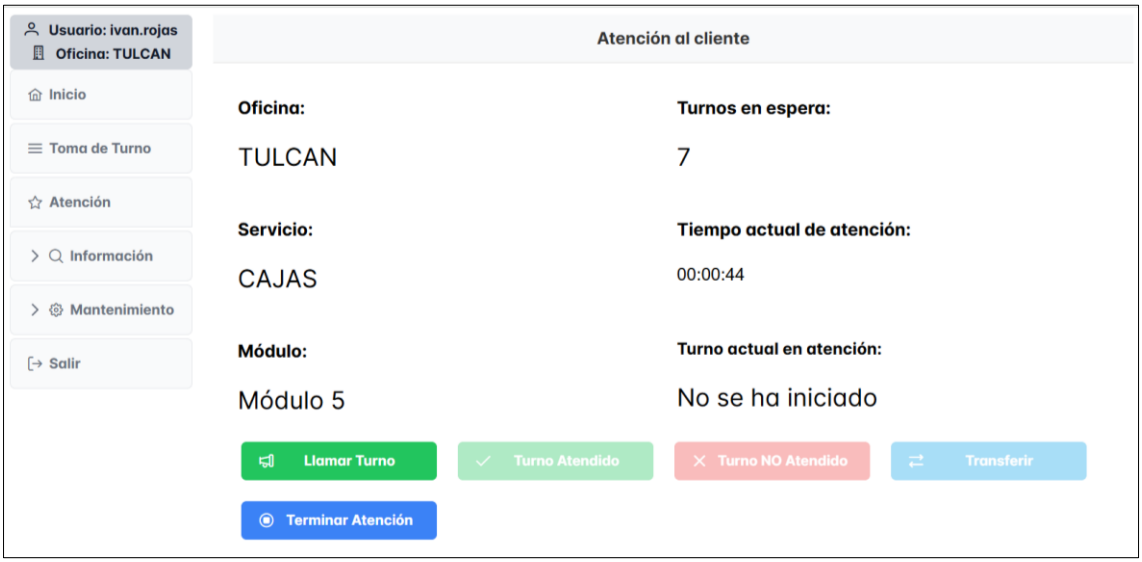
RF21 - Confirmación de turno



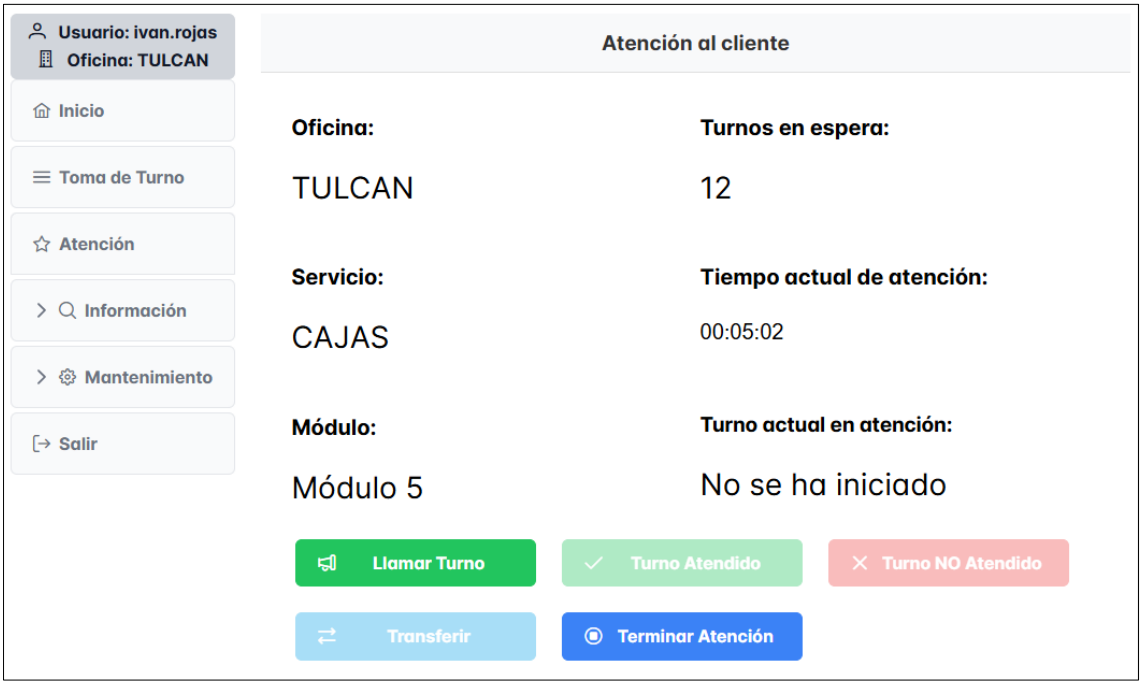
RF21 - Email de confirmación de toma de turno en línea



RNF10 - Diseño responsivo

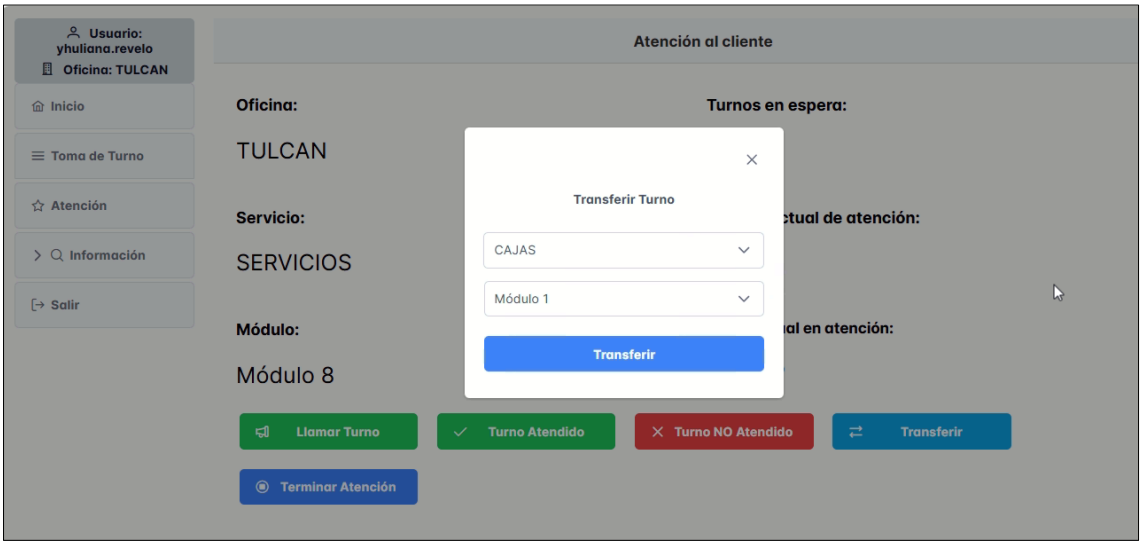


RNF10 – los elementos se ajustan cada tamaño de pantalla

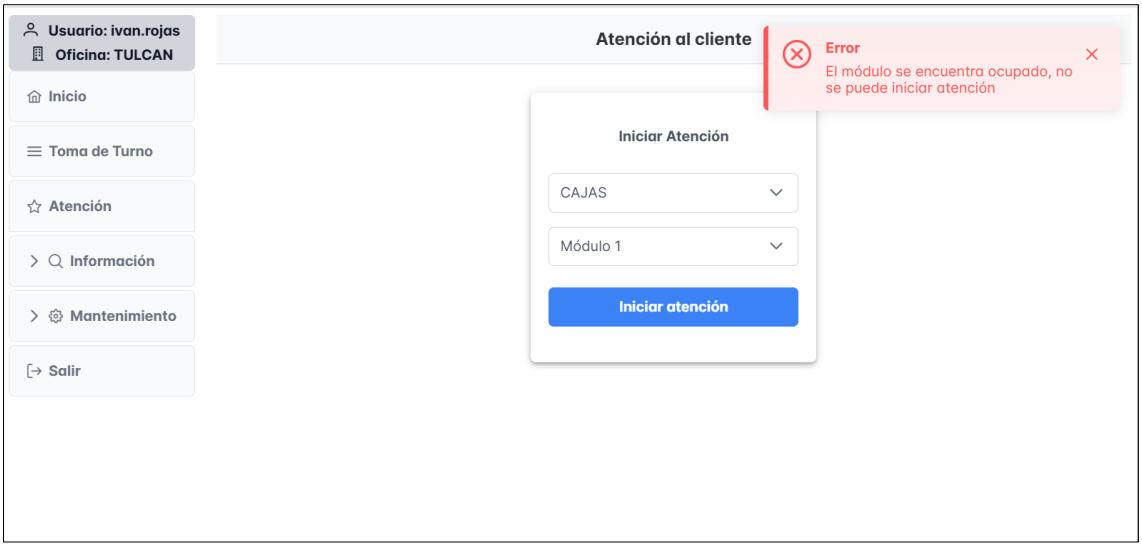


C 5. Sprint 5: Gestión Operativa

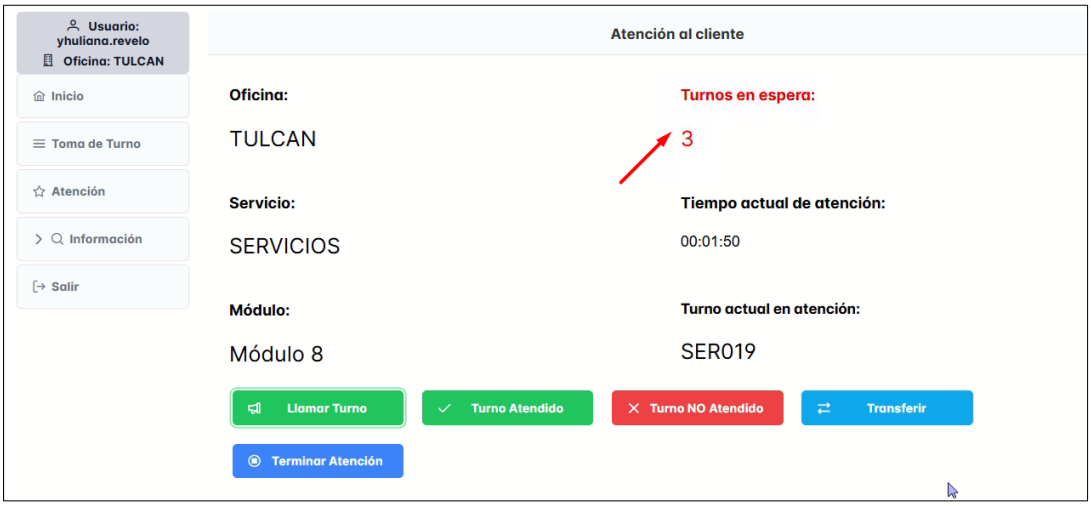
RF06 - Transferencia de turnos a otros módulos



RF16 - Validación de módulo ocupado

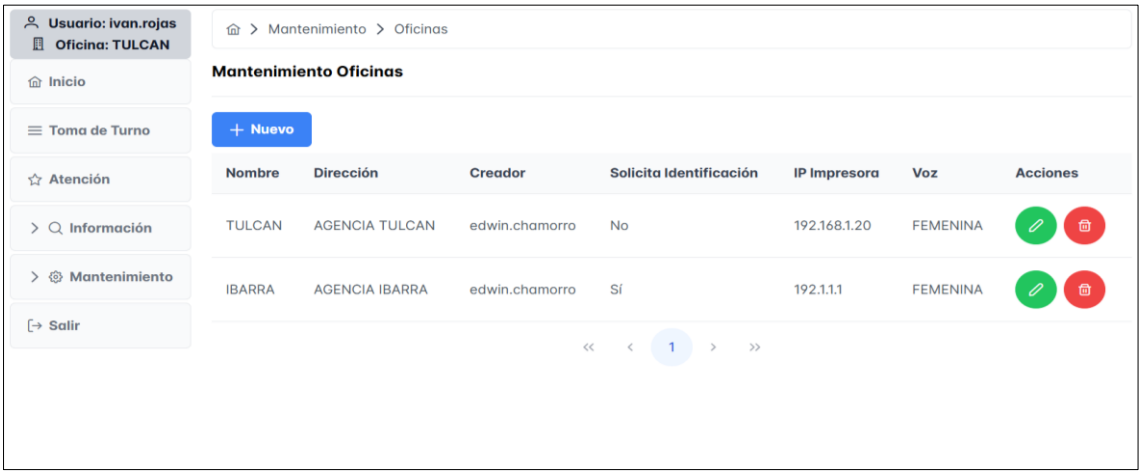


RF22 - Visualización mejorada de turnos en espera

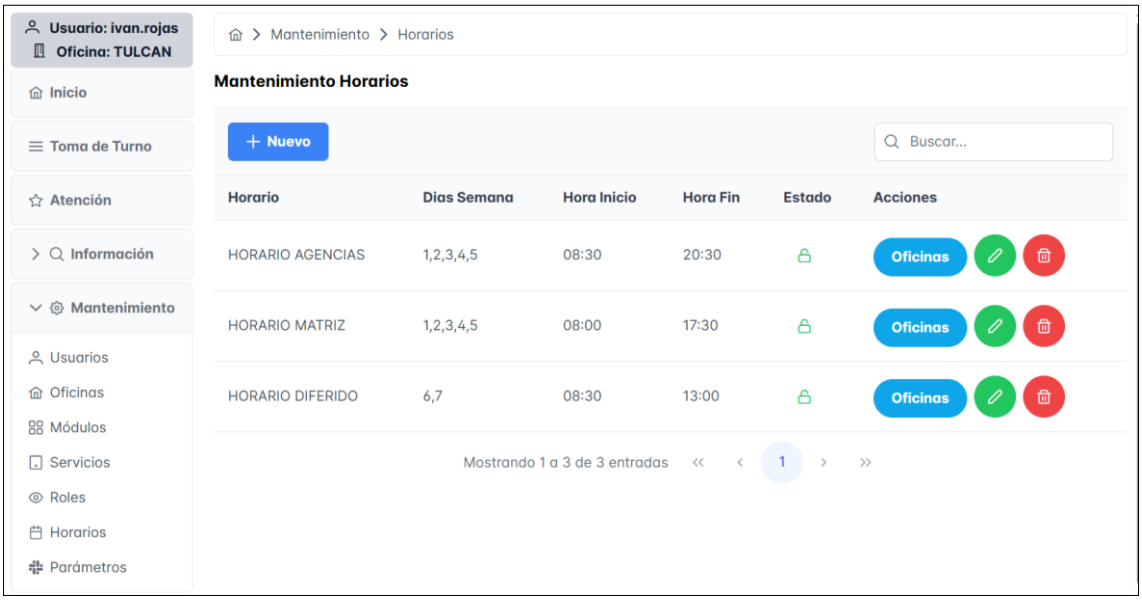


C 6. Sprint 6: Administración Avanzada

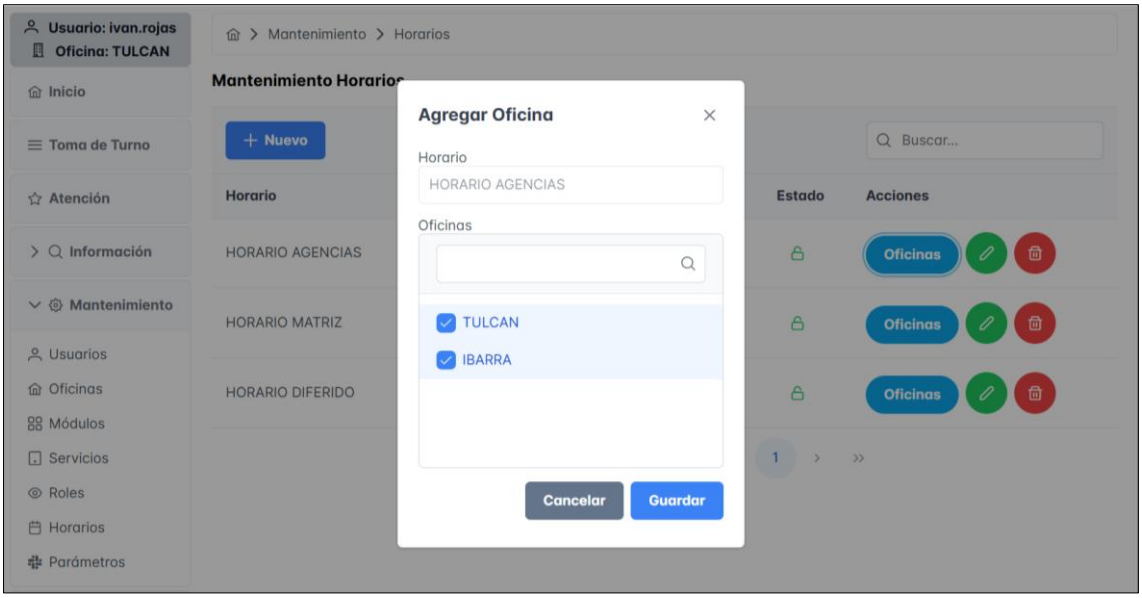
RF09 - Administración de oficinas



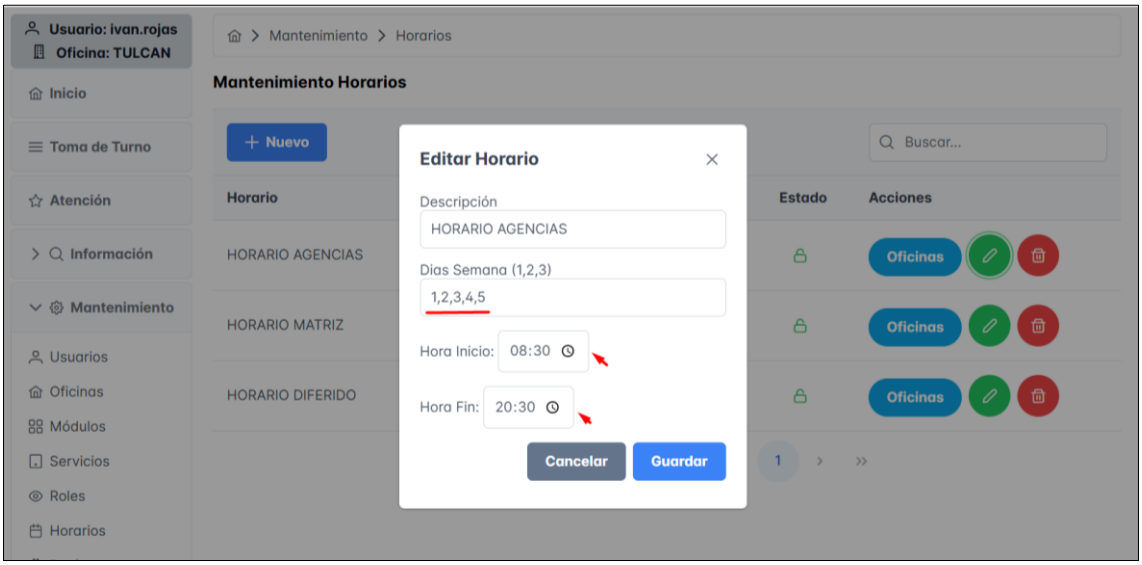
RF11 - Configuración de horarios



RF11 - Parametrización de horarios por cada oficina

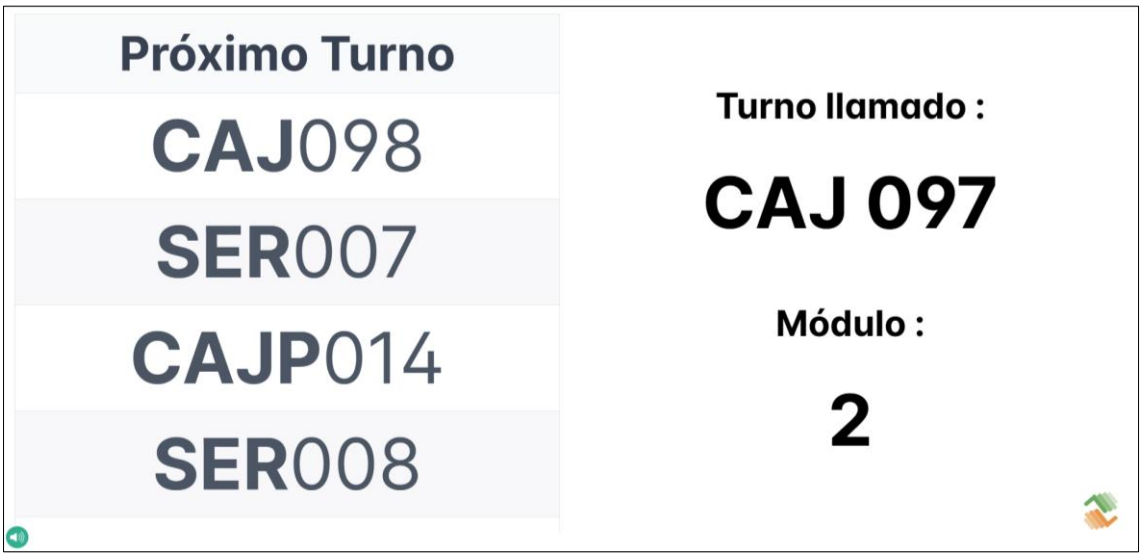


RF26 – Configuración de horarios por días

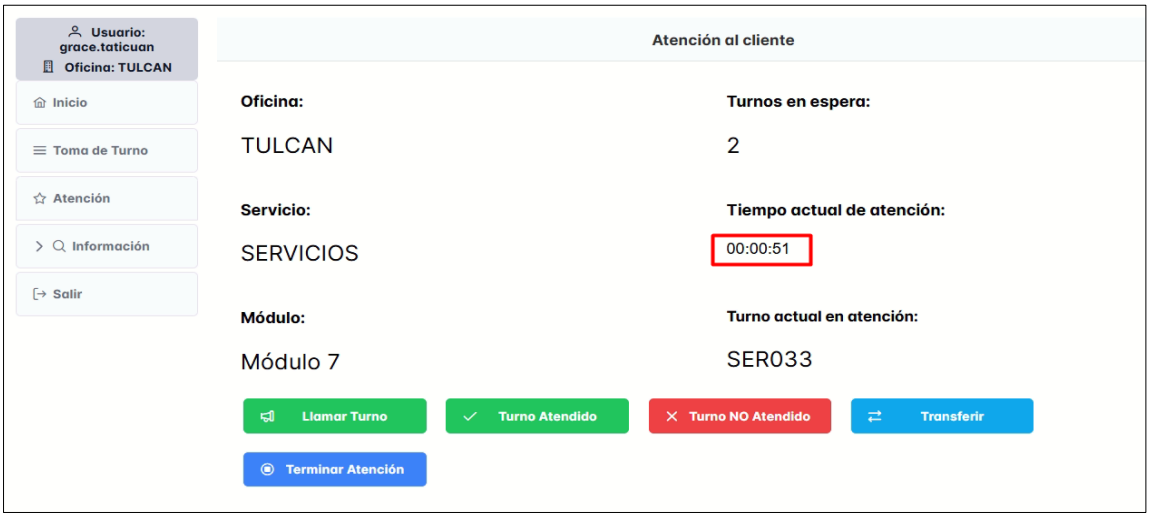


C 7. Sprint 7: Reportes y Estadísticas

RF14 – Actualización de turnos en espera de usuario en tiempo real



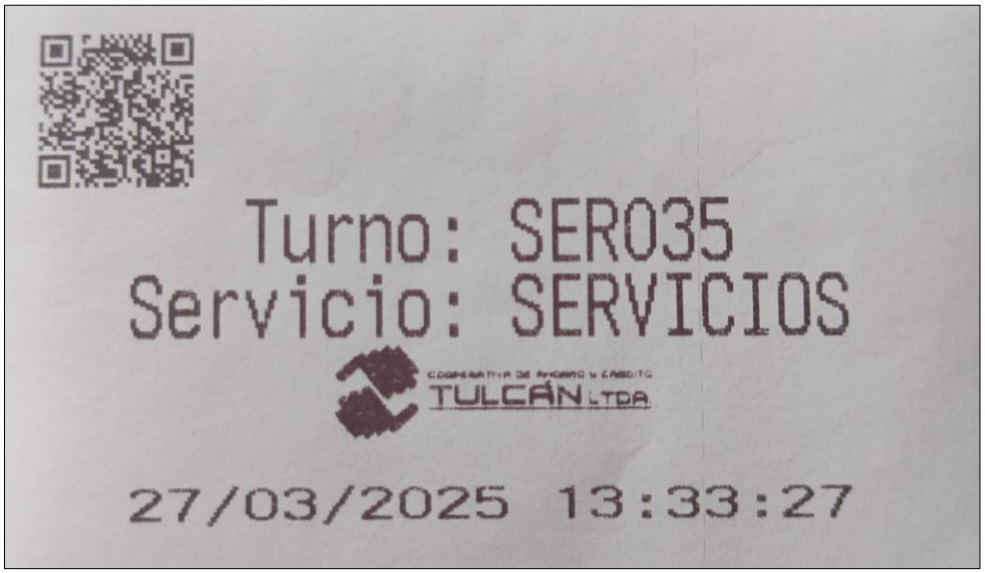
RF15 - Tiempos de atención de turnos llamados



C 8. Sprint 8: Mejoras de Usuario

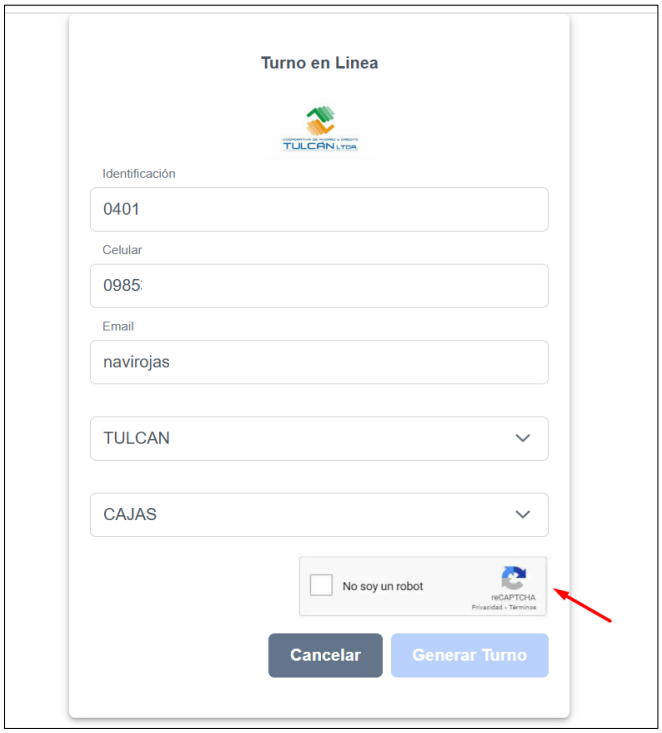
RF14 - Notificaciones audiovisuales

RF18 - Ticket con QR

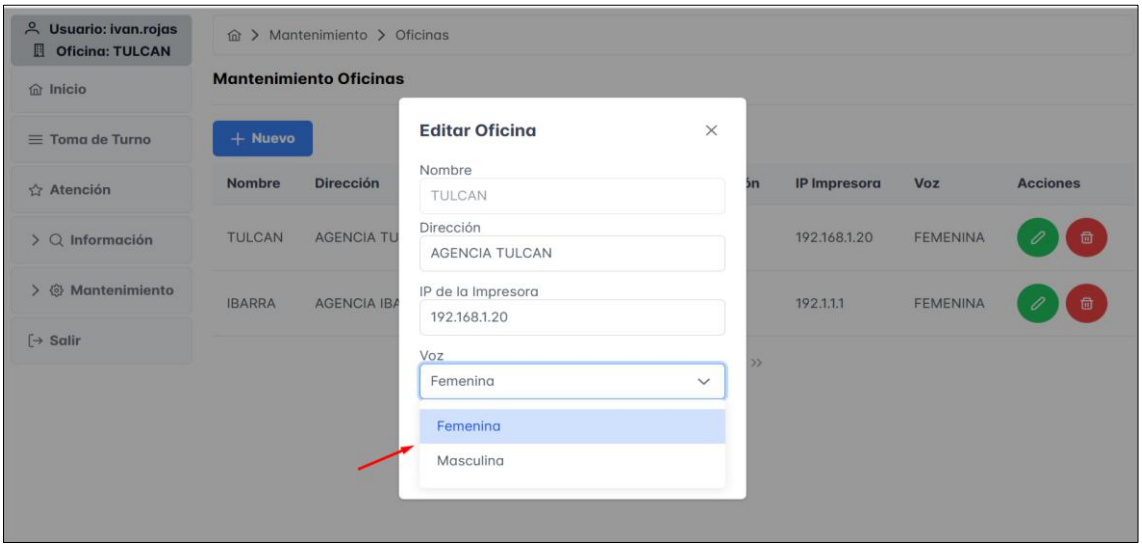


C 9. Sprint 9: Seguridad y Validaciones

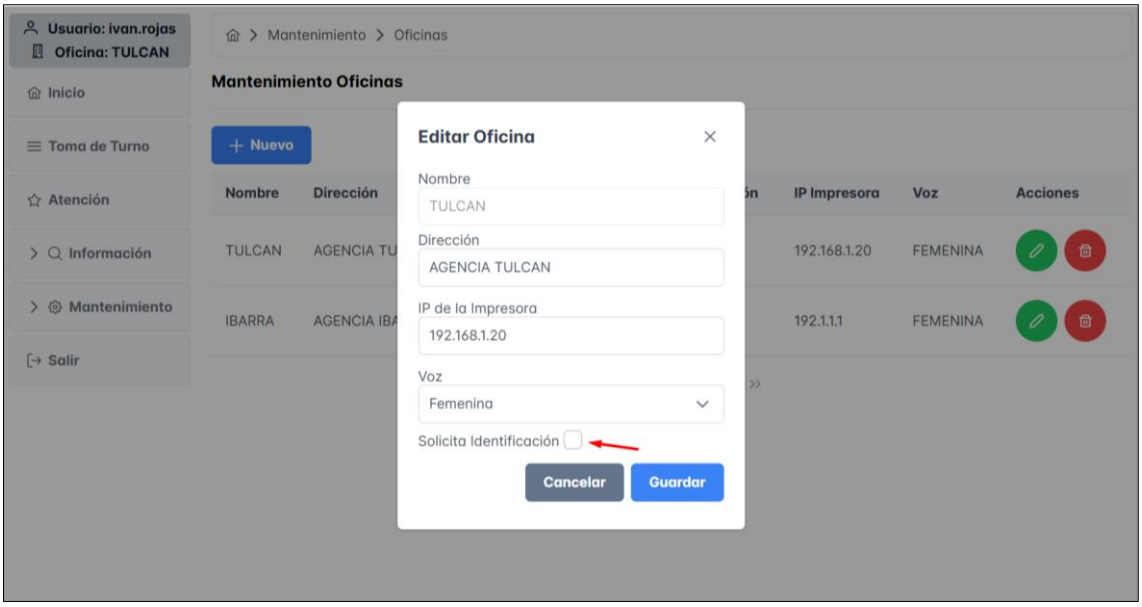
RF20 – Sistema de seguridad CAPTCHA



RF23 – Parametrización de voz en el sistema



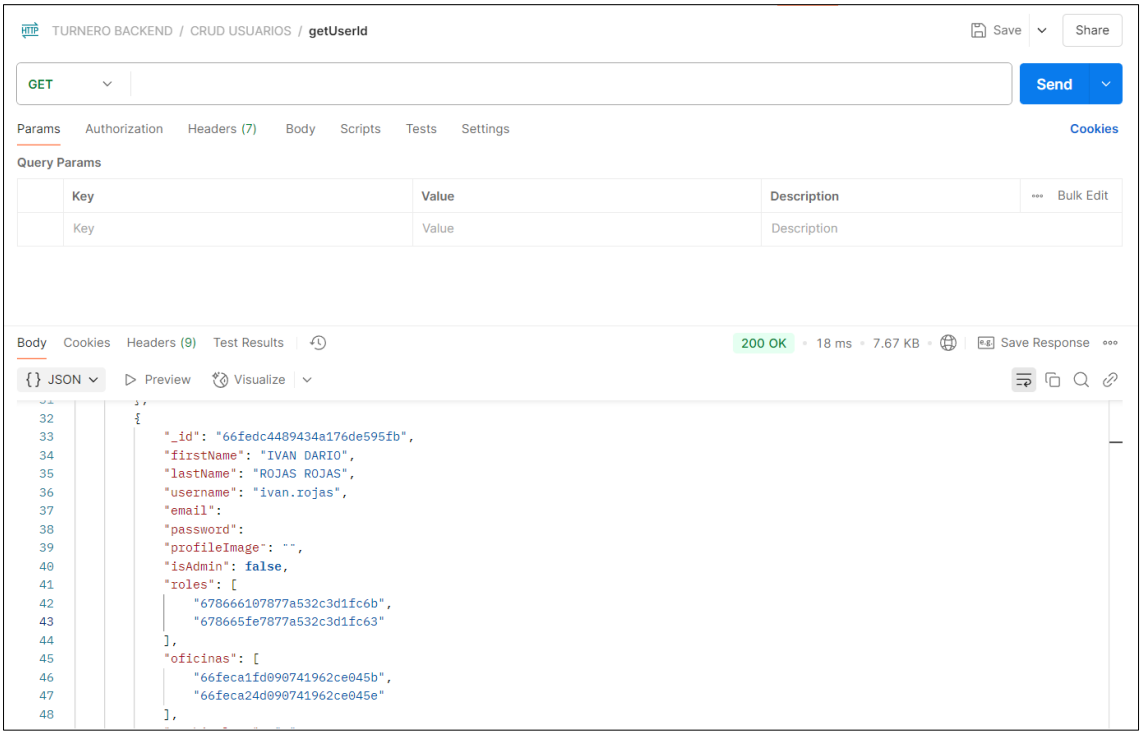
RF24 – Configuración de validación de identidad al tomar turno



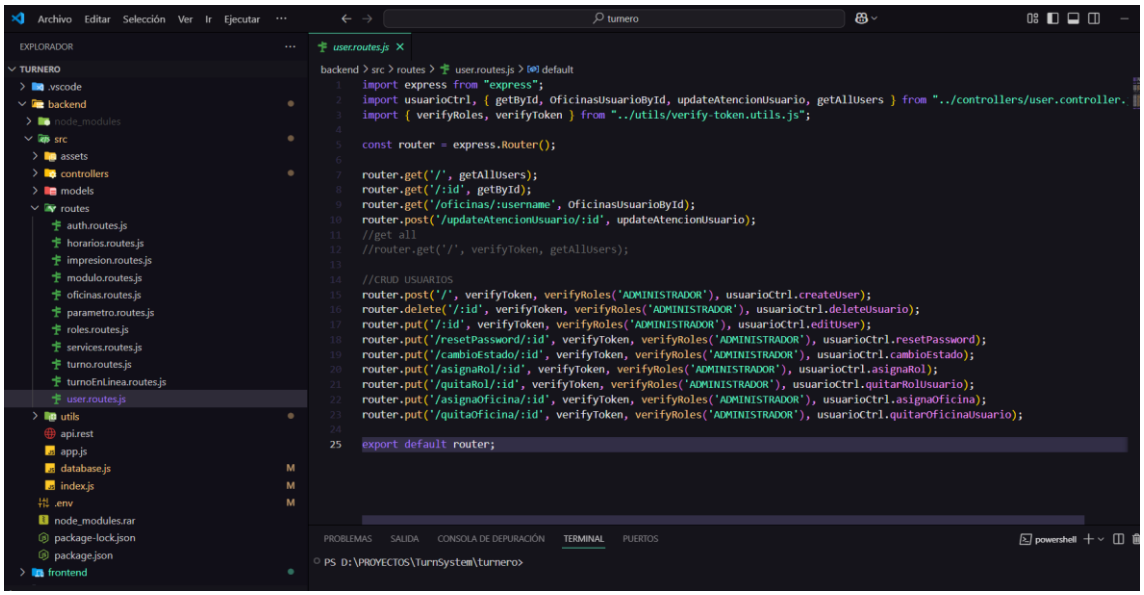
RF24 - Solicitud de identificación al tomar turno

C 10. Sprint 10: Optimizaciones Finales

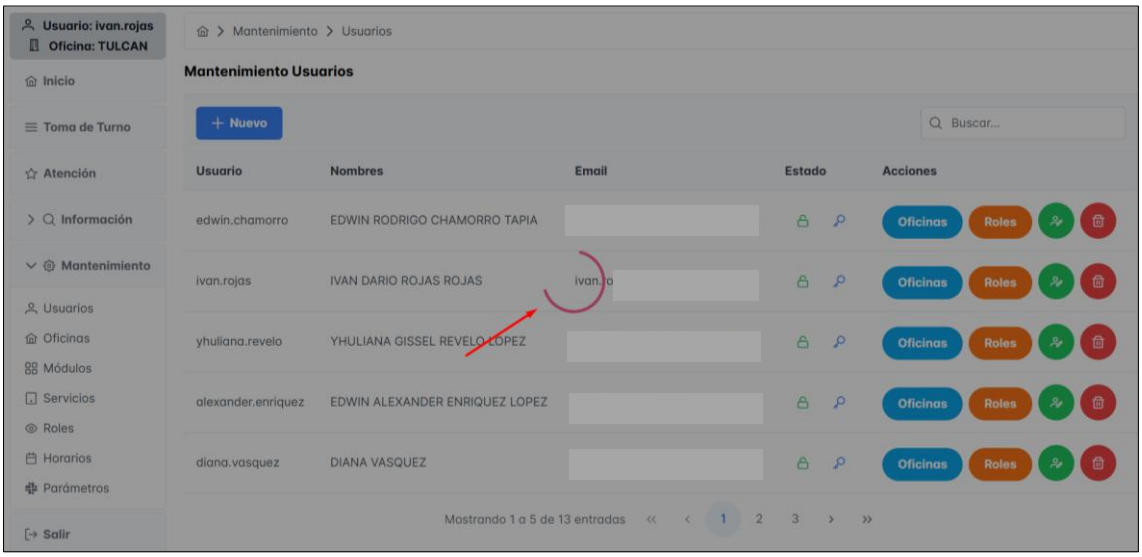
RNF03 - API's de integración al sistema



RNF03 - Esquema de rutas en el backend



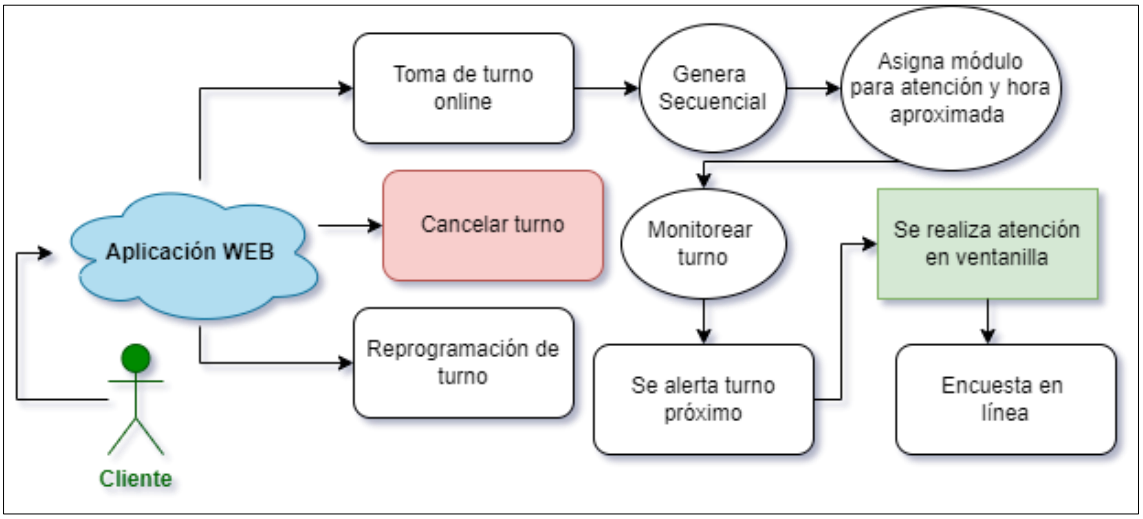
RNF14 - Indicadores visuales al momento de realizar



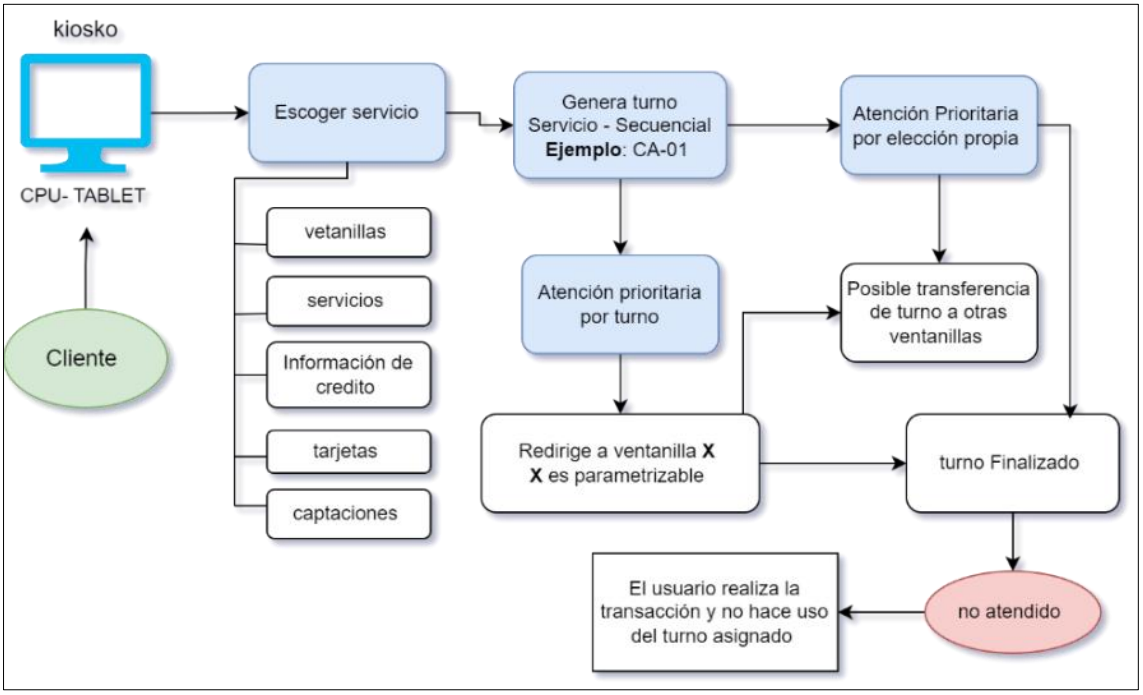
ANEXO D. Marco tecnológico

D 1. Diagramas de flujo propuestos para el sistema

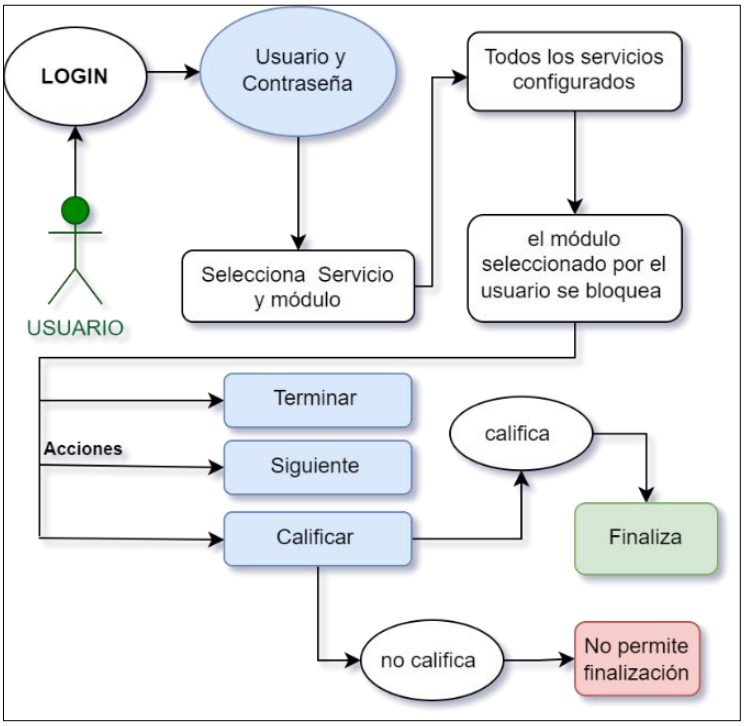
Flujo de toma de turnos en línea



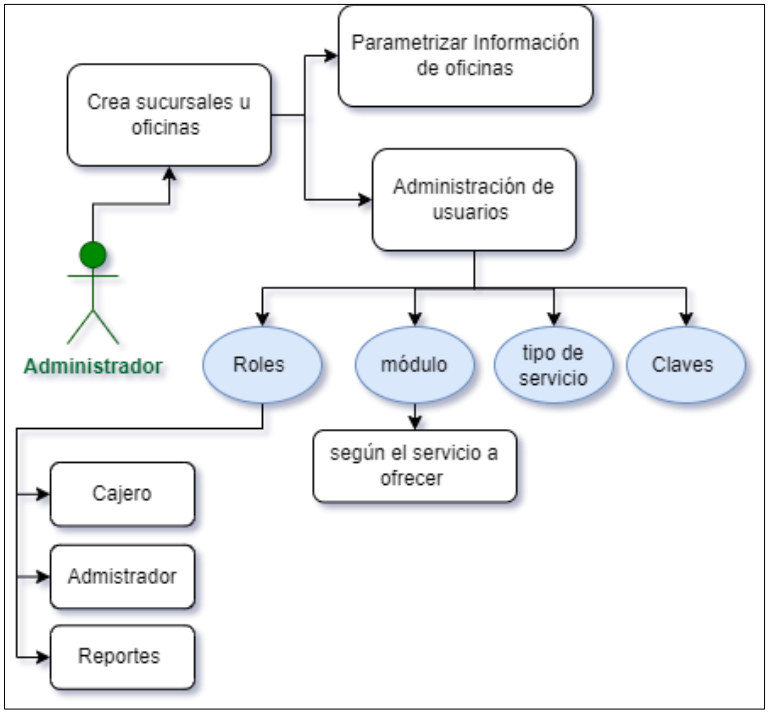
Flujo de toma de turnos presenciales



Flujo de atención al cliente

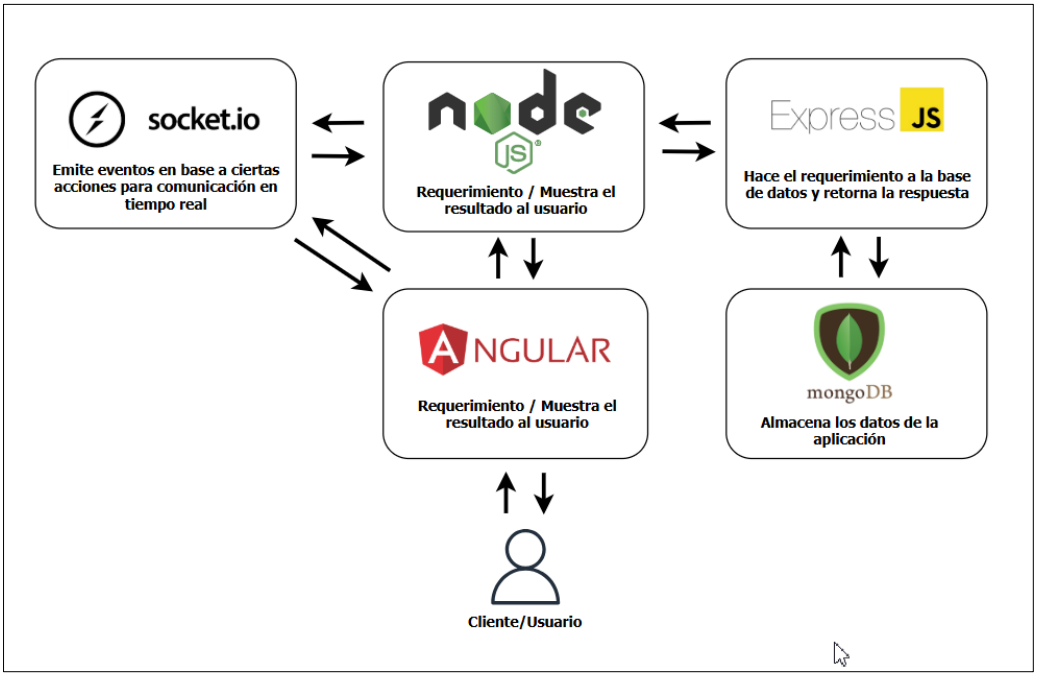


Flujo administrativo

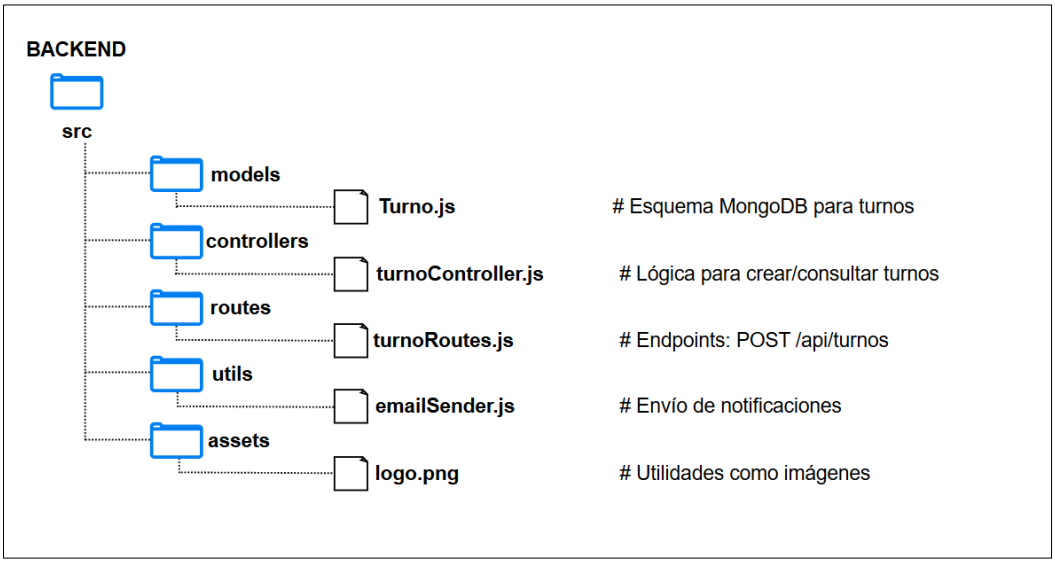


D 2. Arquitectura

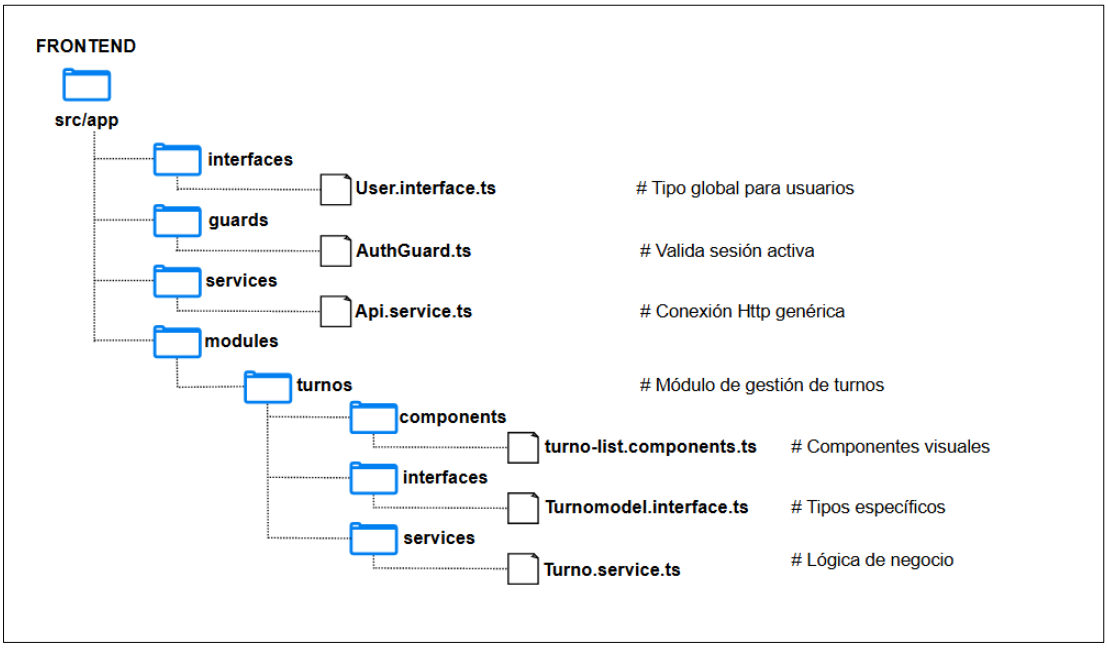
Arquitectura de despliegue



Arquitectura de aplicación backend

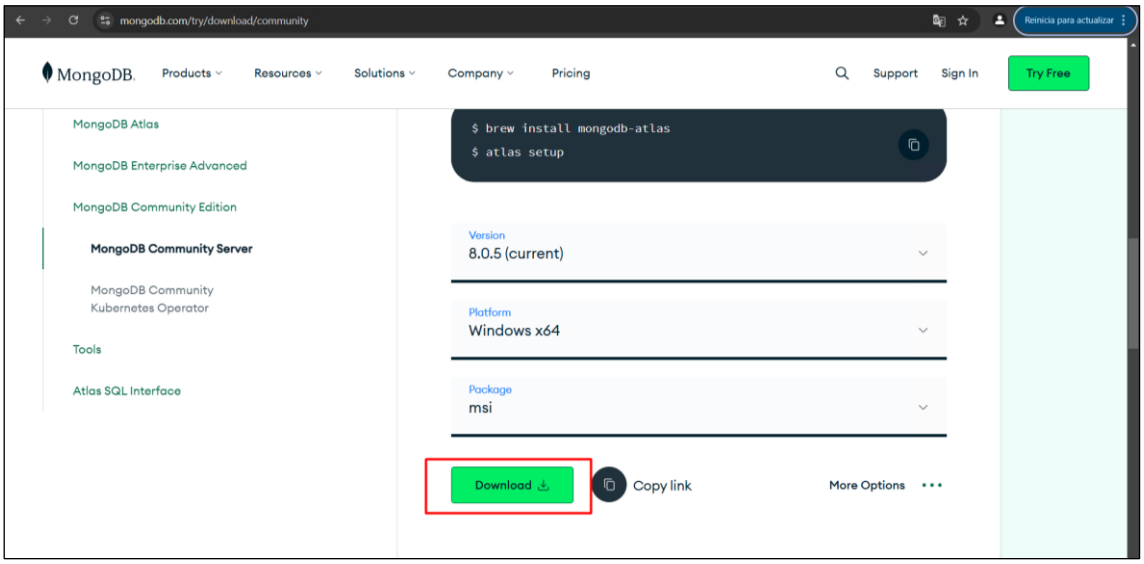


Arquitectura de aplicación frontend

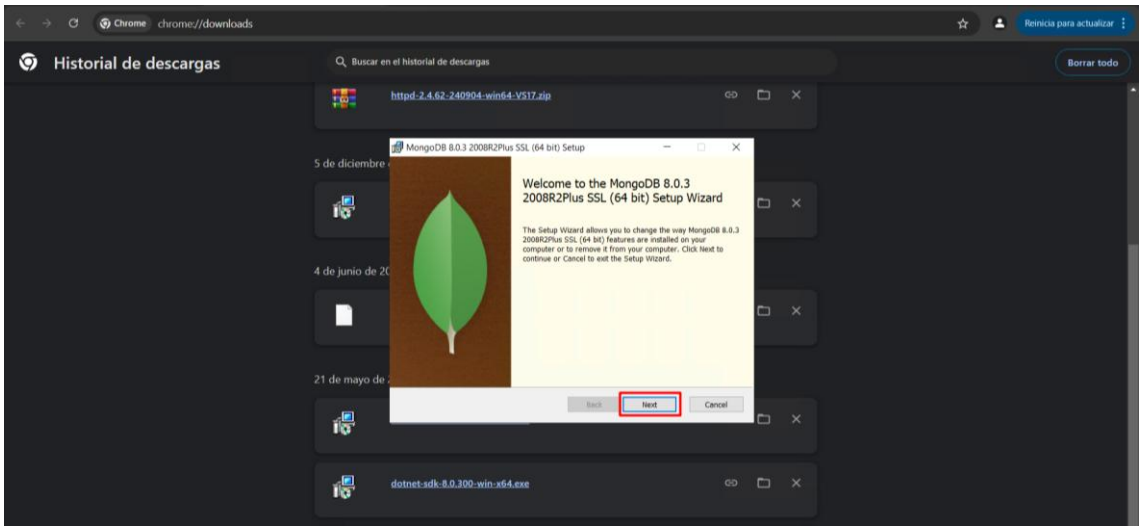


D 3. Despliegue en servidor

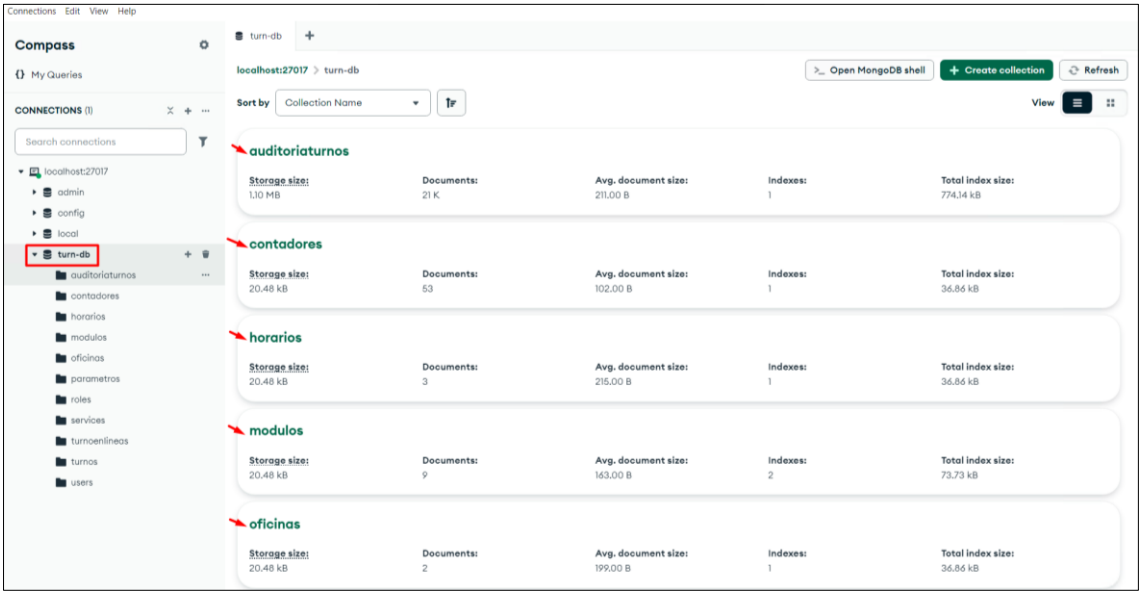
Instalación de MongoDB en servidor



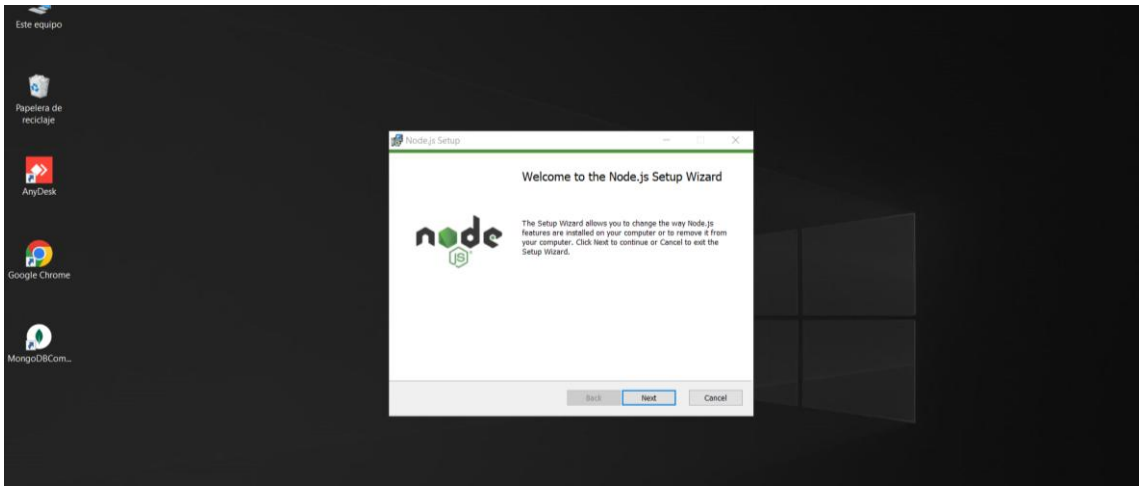
Configuración de MongoDB y MongoCompass



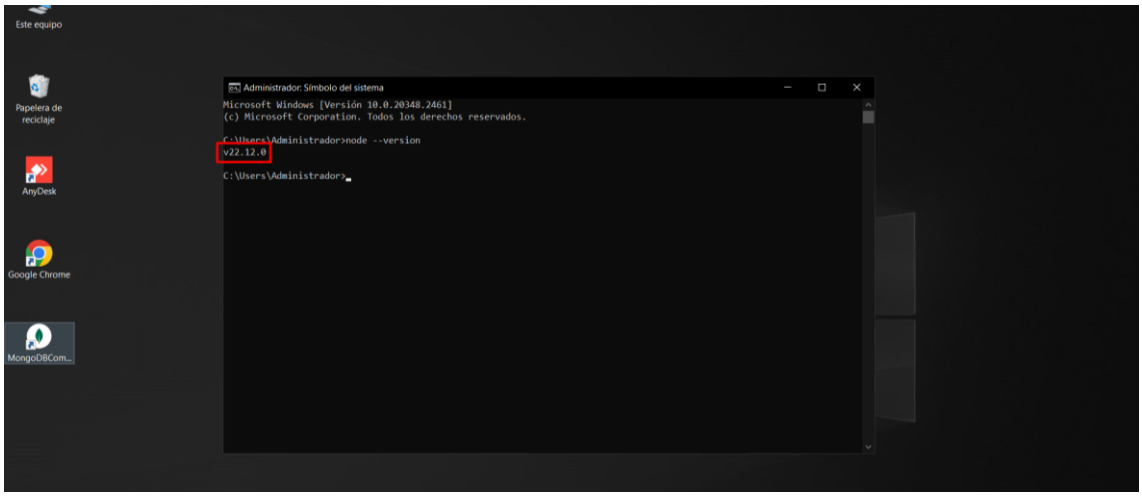
Creación de base de datos en MongoCompass



Instalación de Node.js



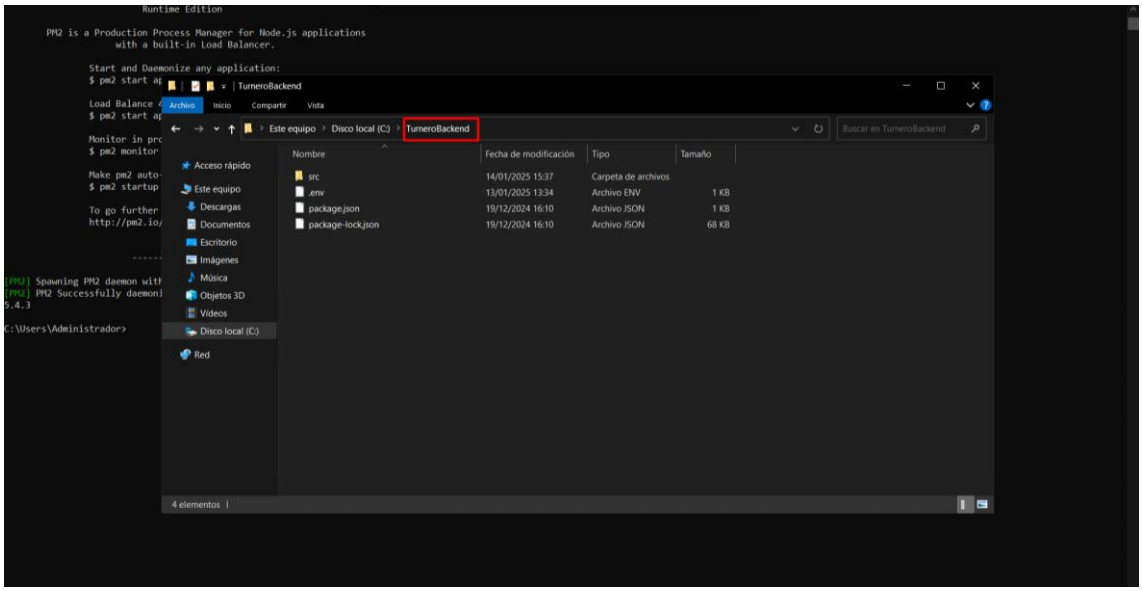
Verificación de la correcta instalación de Node.js



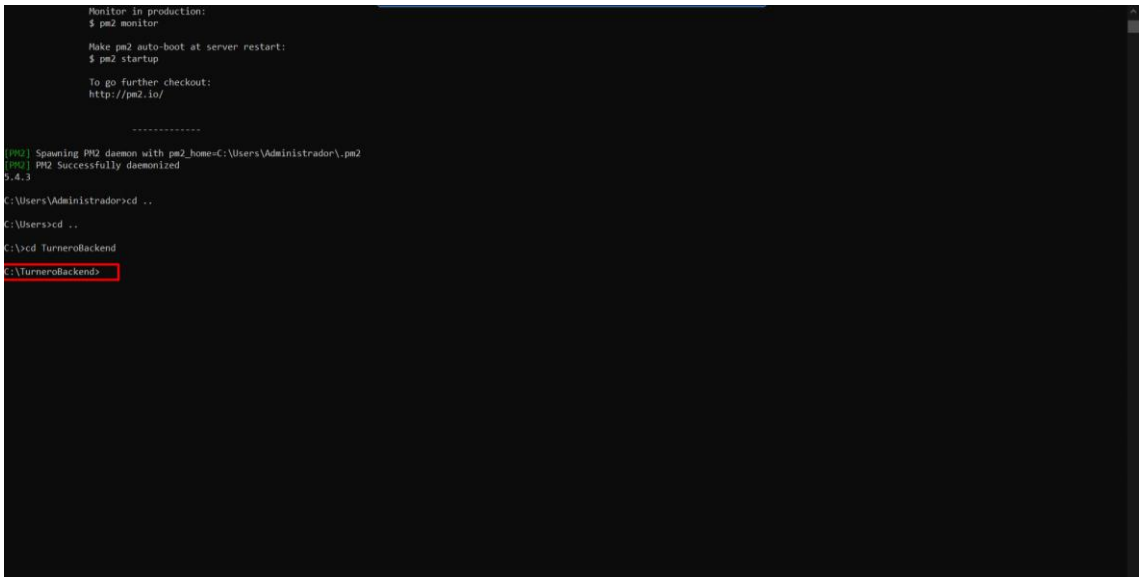
Instalación de Módulo de Gestión de Procesos (PM2)

Verificación de instalación de Módulo de Gestión de Procesos (PM2)

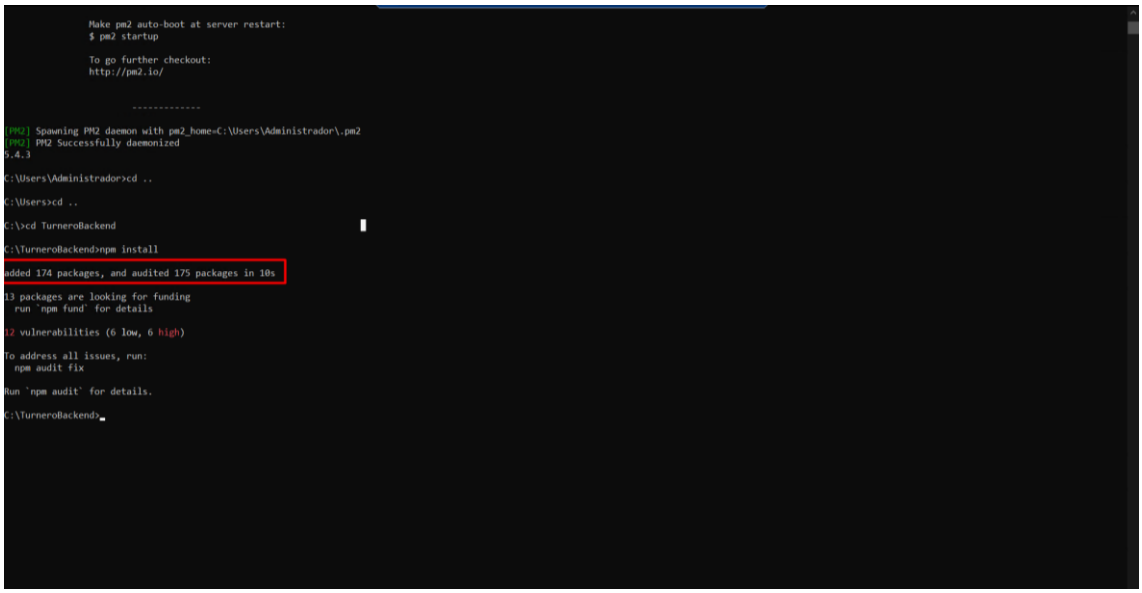
Copia del proyecto en servidor de despliegue



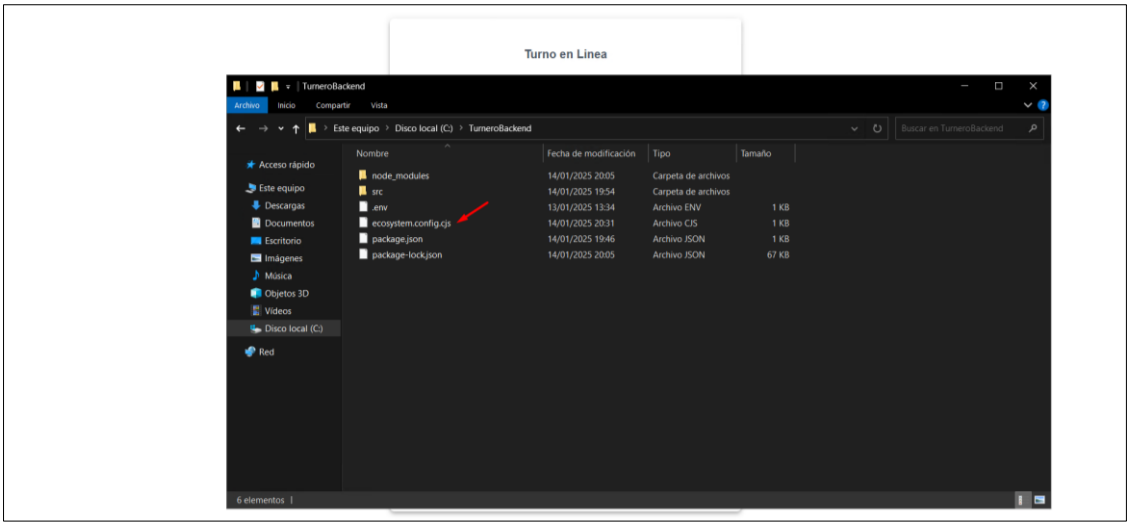
Ruta del proyecto en el servidor de despliegue



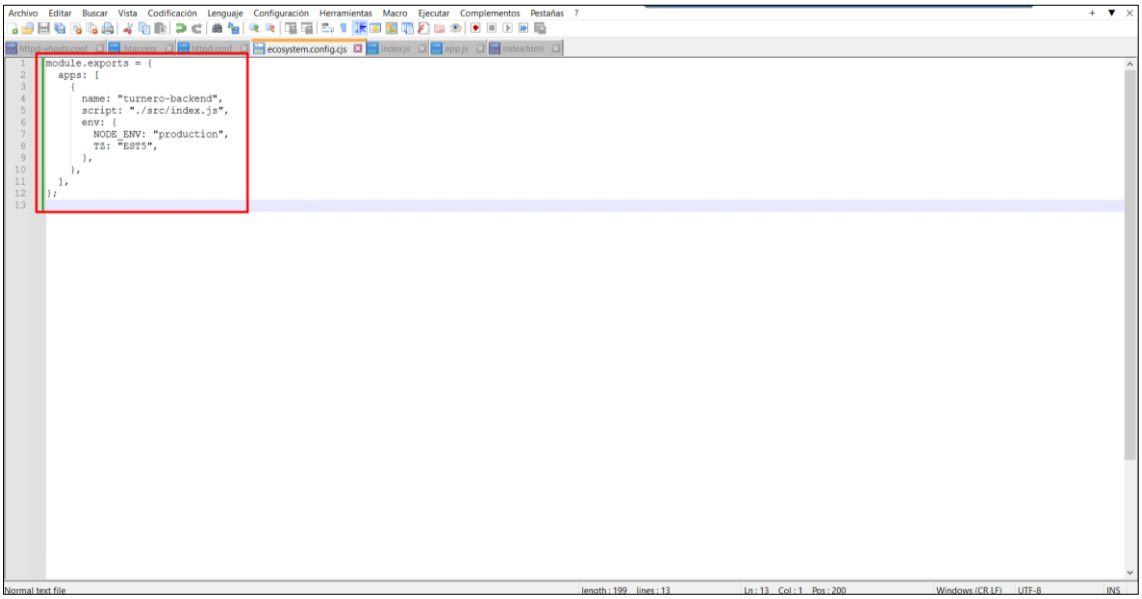
Instalación de dependencias del proyecto en el servidor



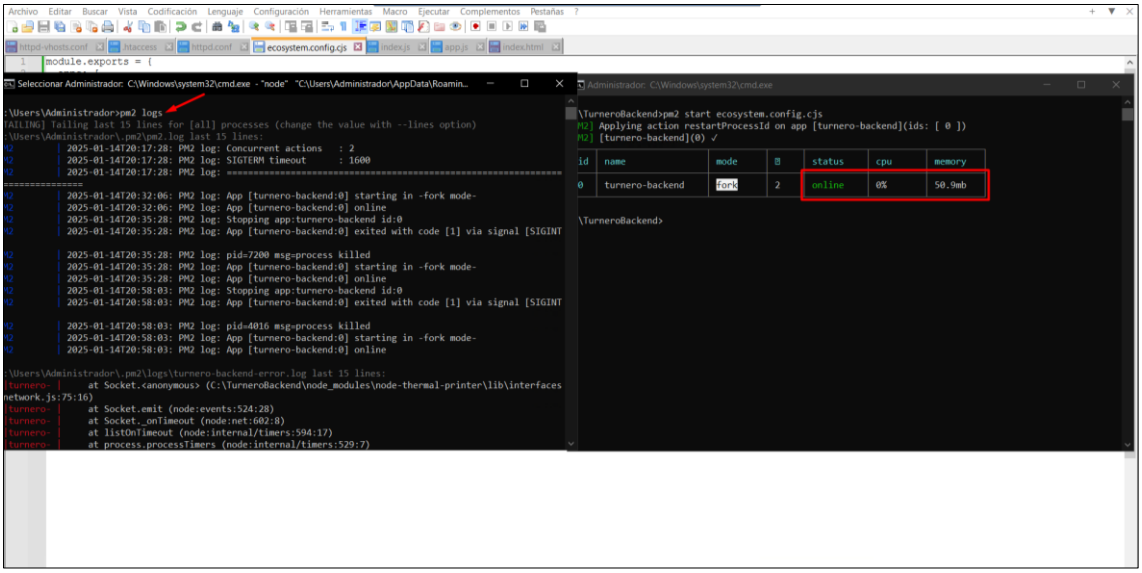
Archivo de configuración para PM2



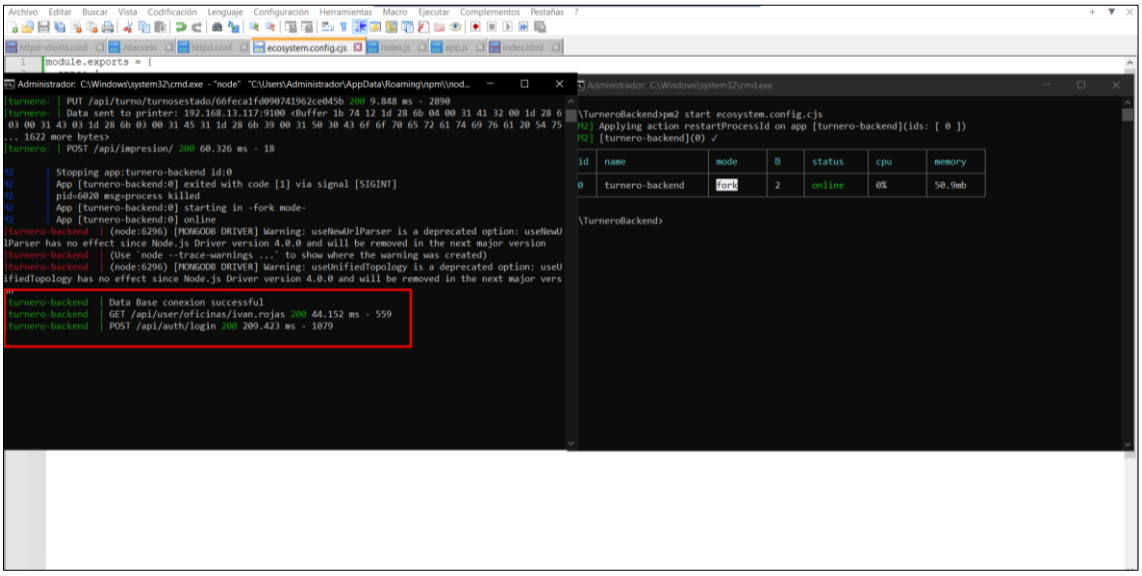
Contenido de archivo para inicio de PM2 ecosystem.config.cjs



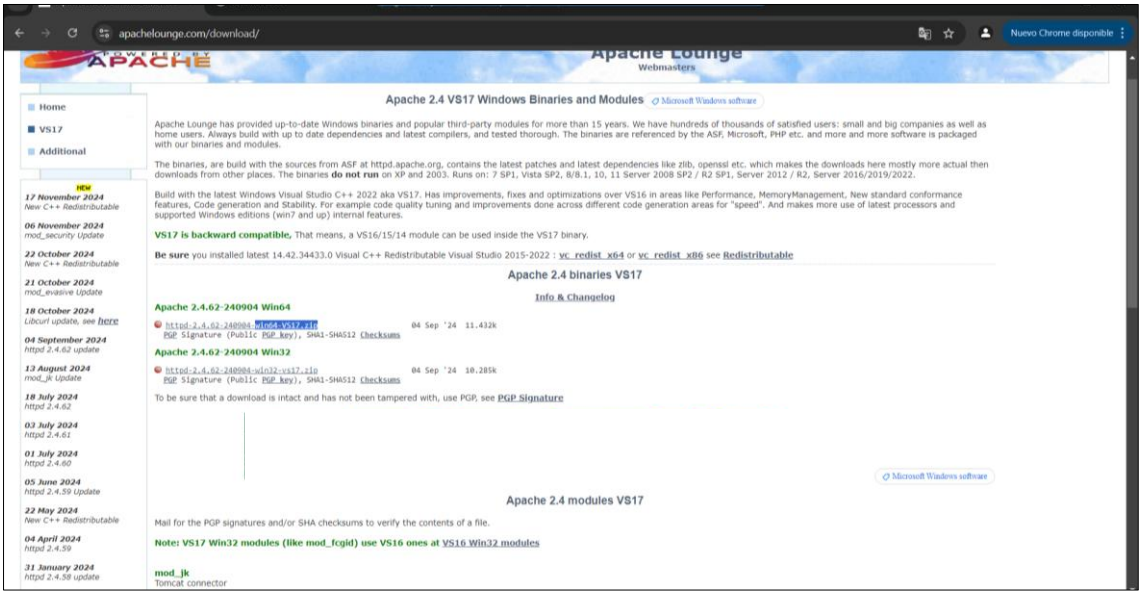
Monitoreo de logs de backend con ayuda de PM2



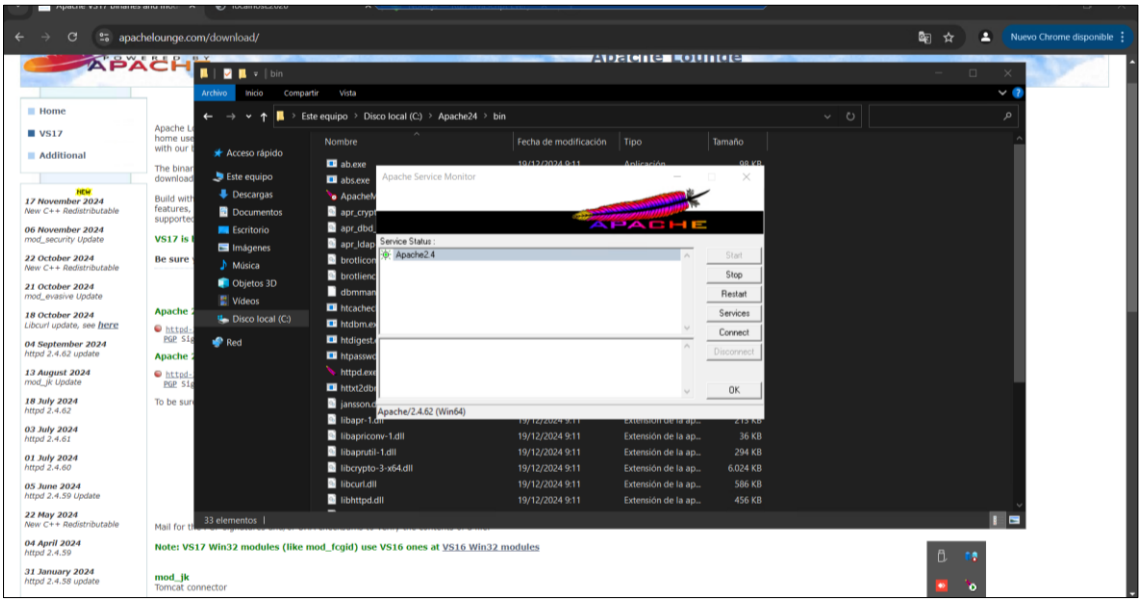
Levantamiento del proceso del backend y conexión a la base de datos



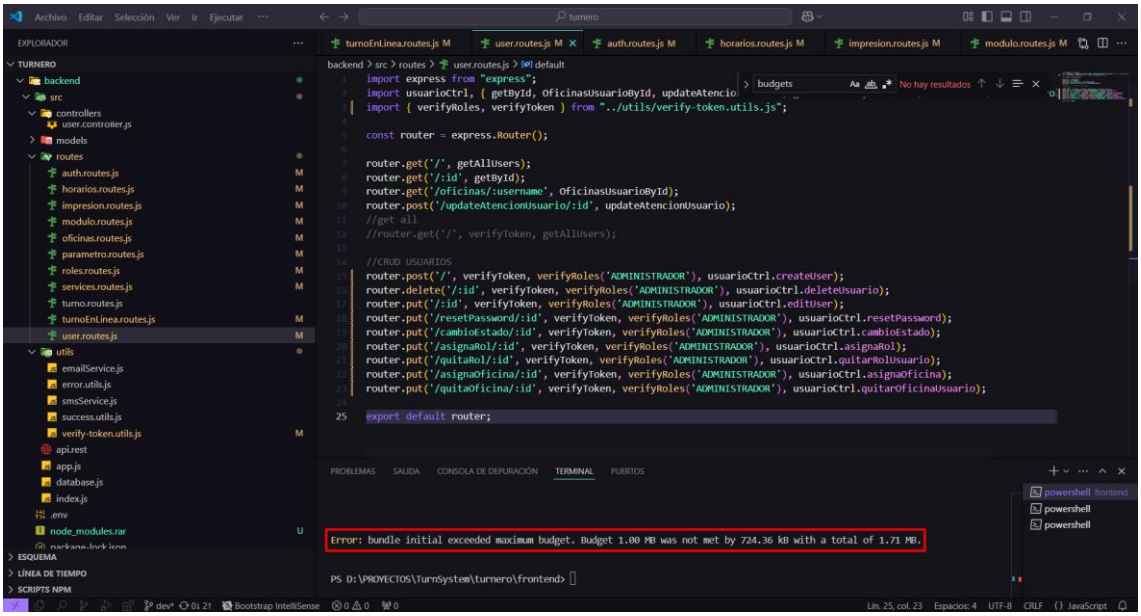
Versión instalada de servidor de aplicaciones Apache



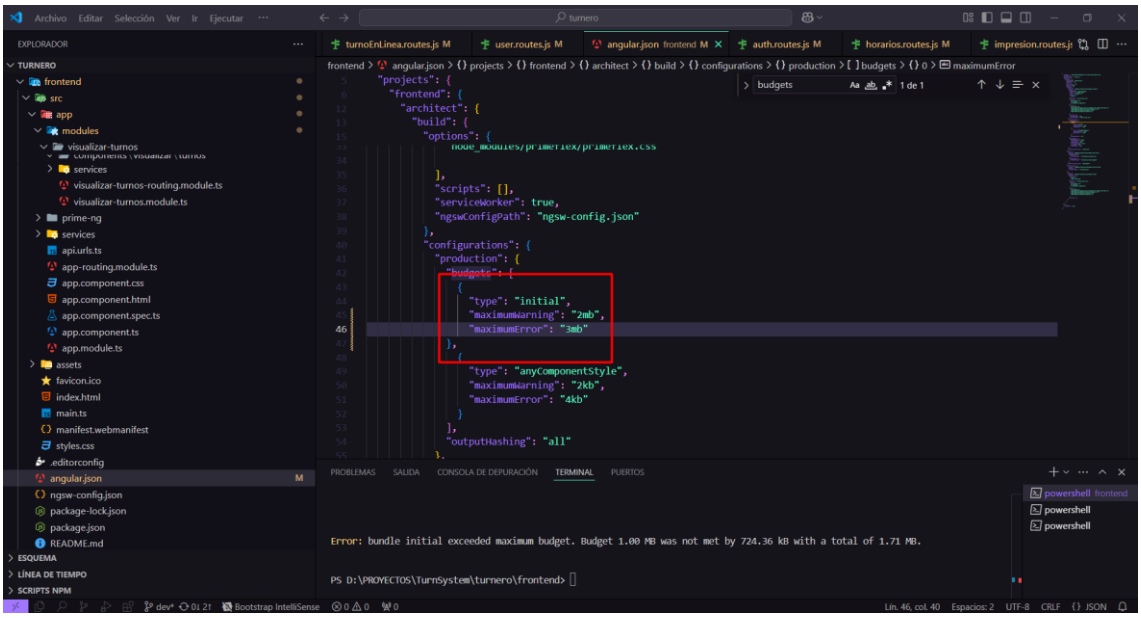
Inicio de servidor de aplicaciones Apache



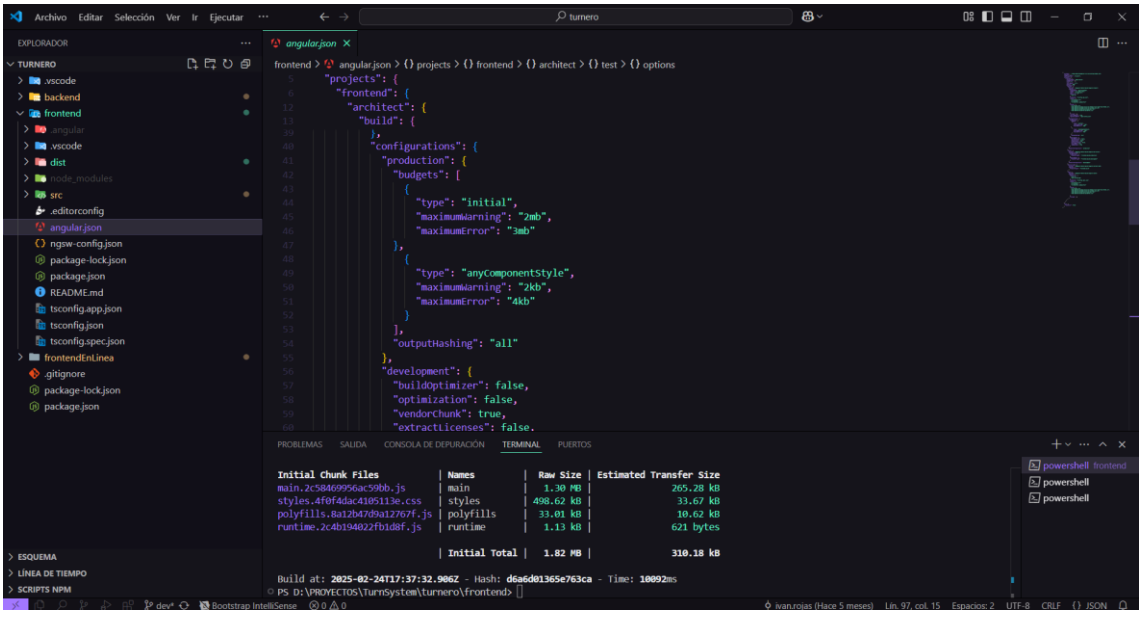
Error al generar build de Angular



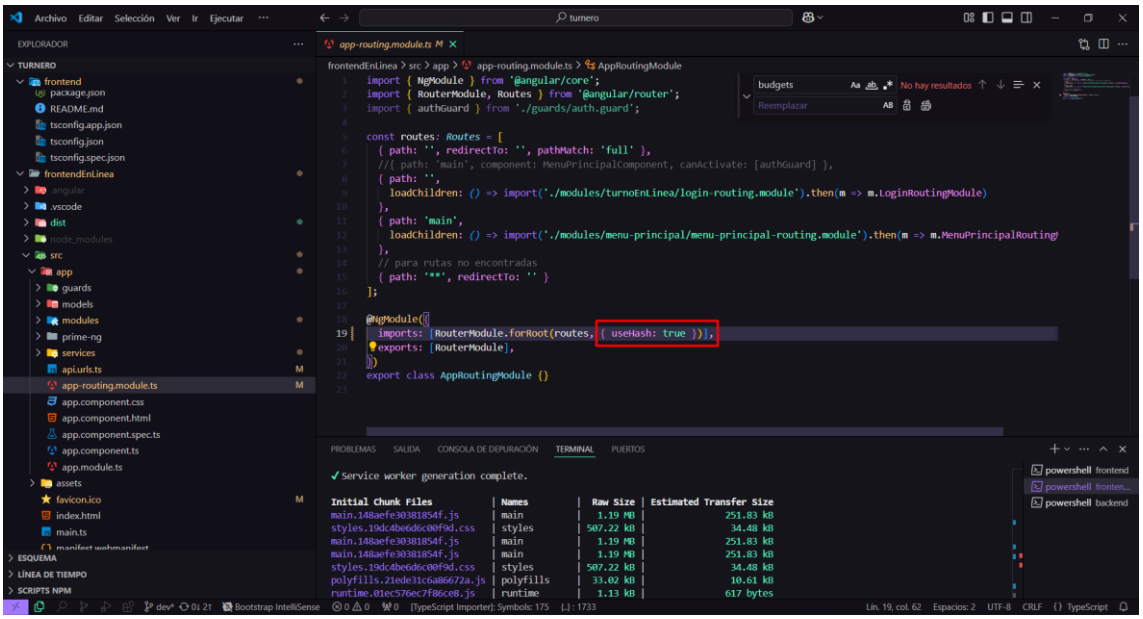
Modificación de angular.json para corrección de error



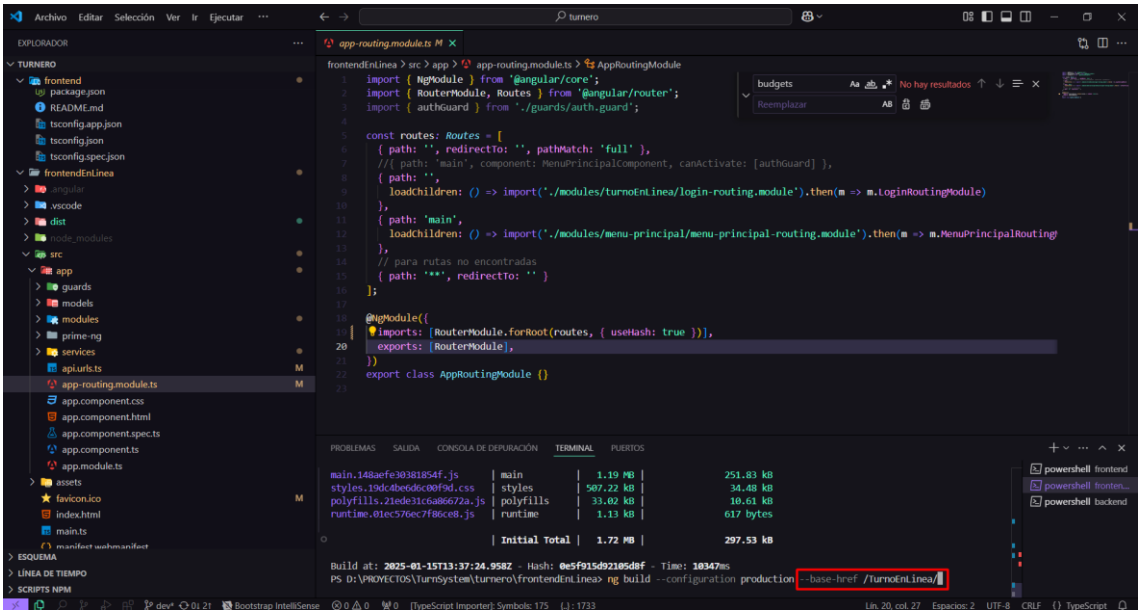
Generación de build exitoso en Angular



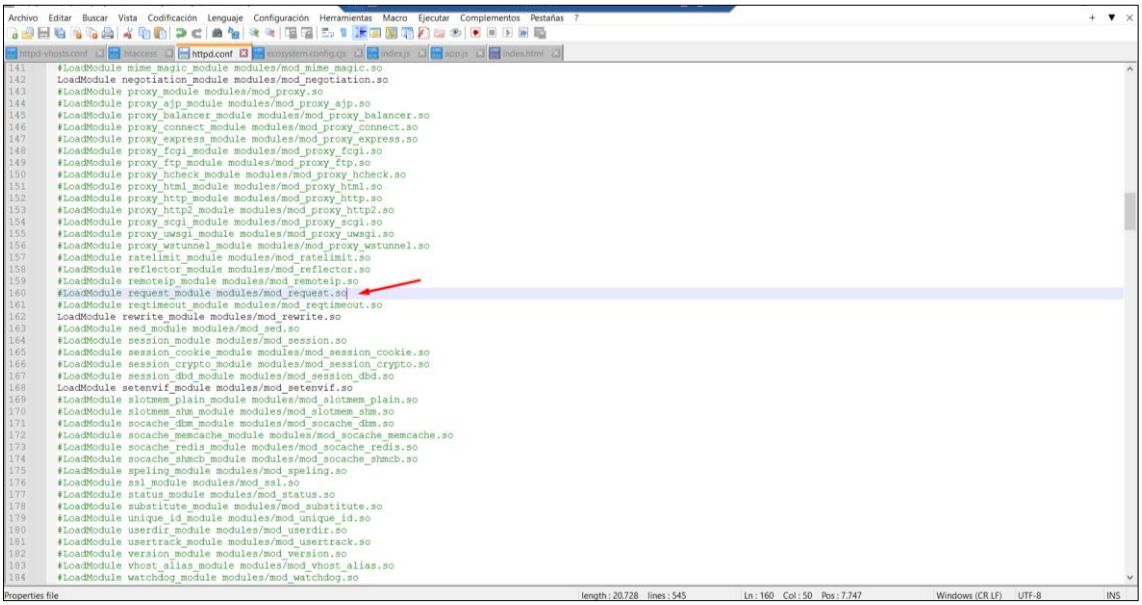
Configuración de archivo app-routing.js para inconveniente de rutas



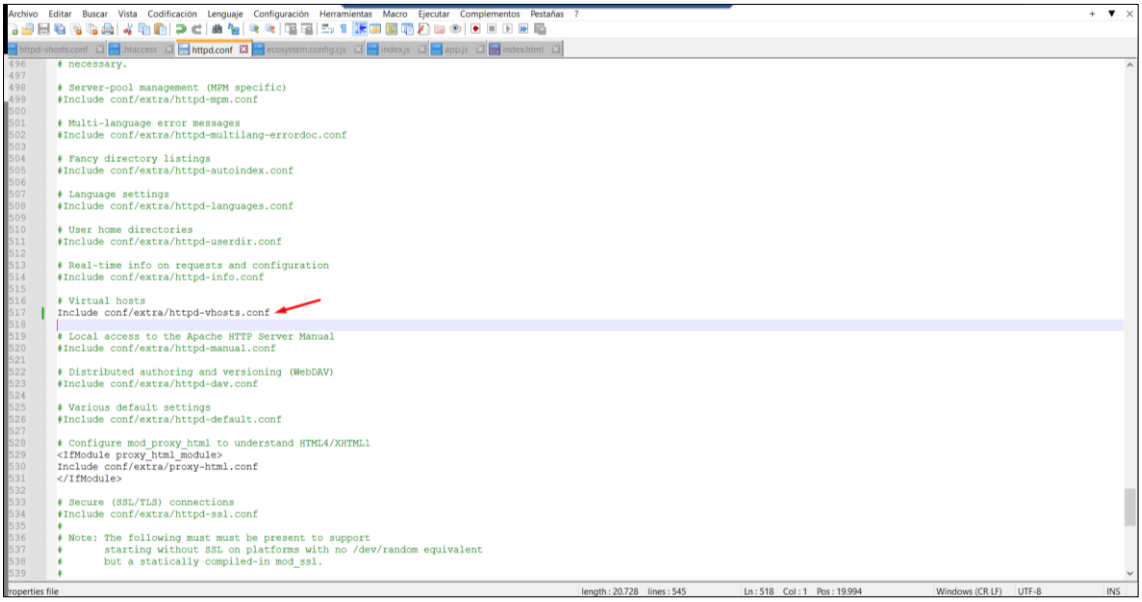
Generación de Build en Angular con ruta base



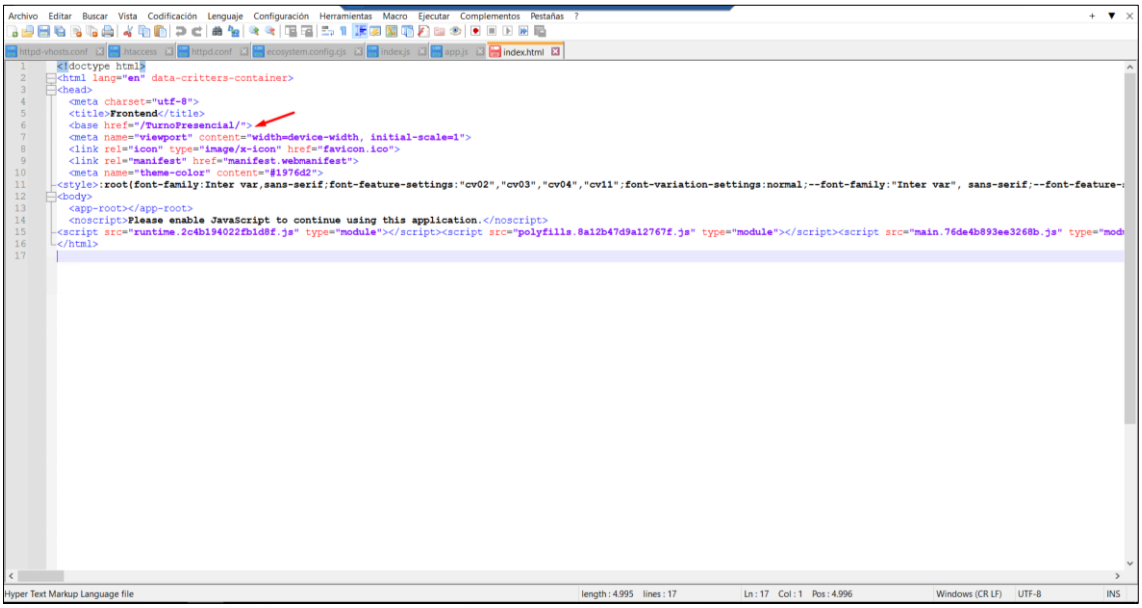
Modificación del archivo de configuración de Apache httpd.conf



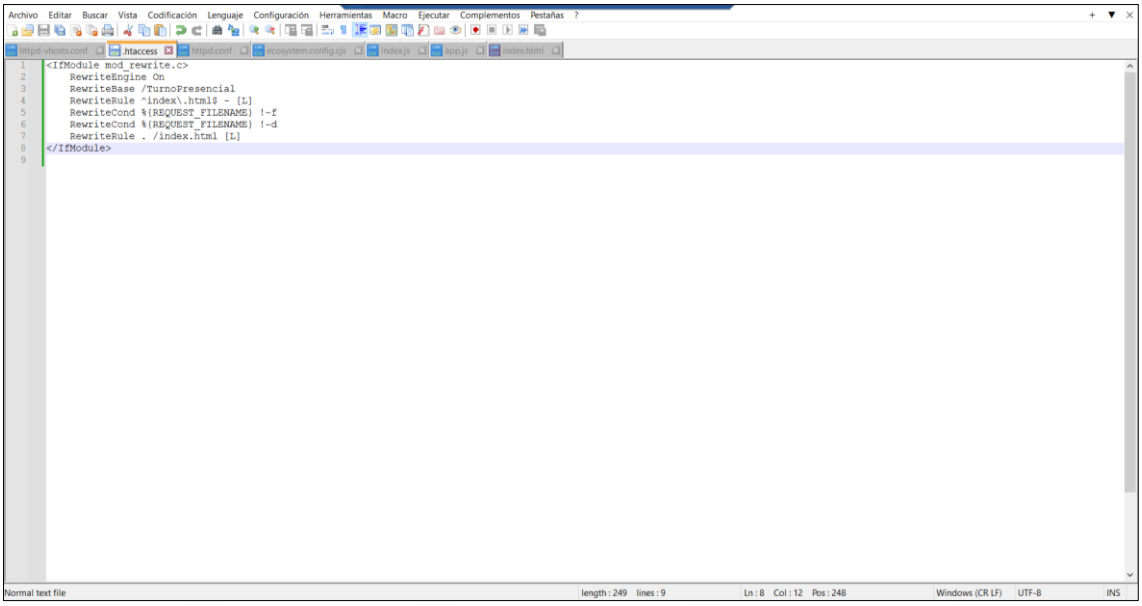
Configuración de adicional de httpd.conf



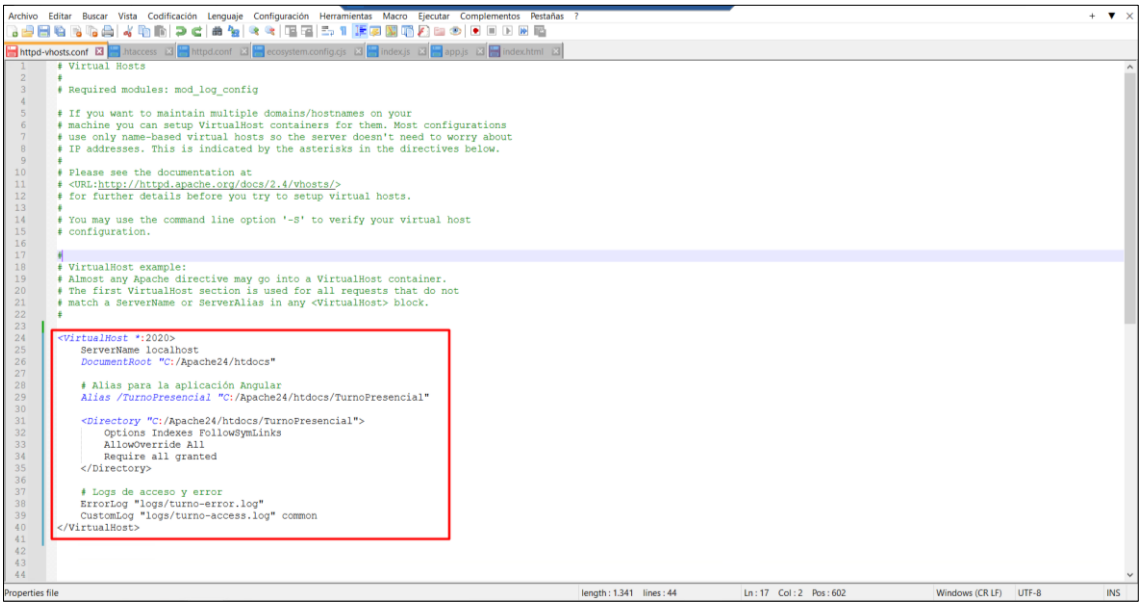
Configuración requerida en archivo de proyecto en Angular index.html



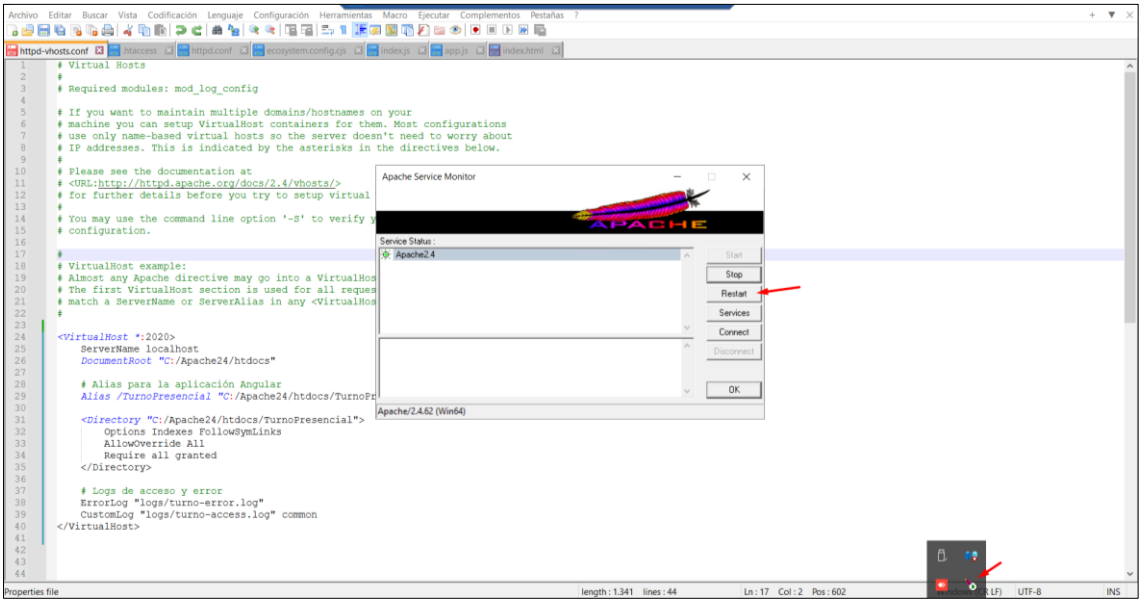
Creación y contenido de archivo .htaccess



Configuración de archivo de Apache httpd-vhosts.conf

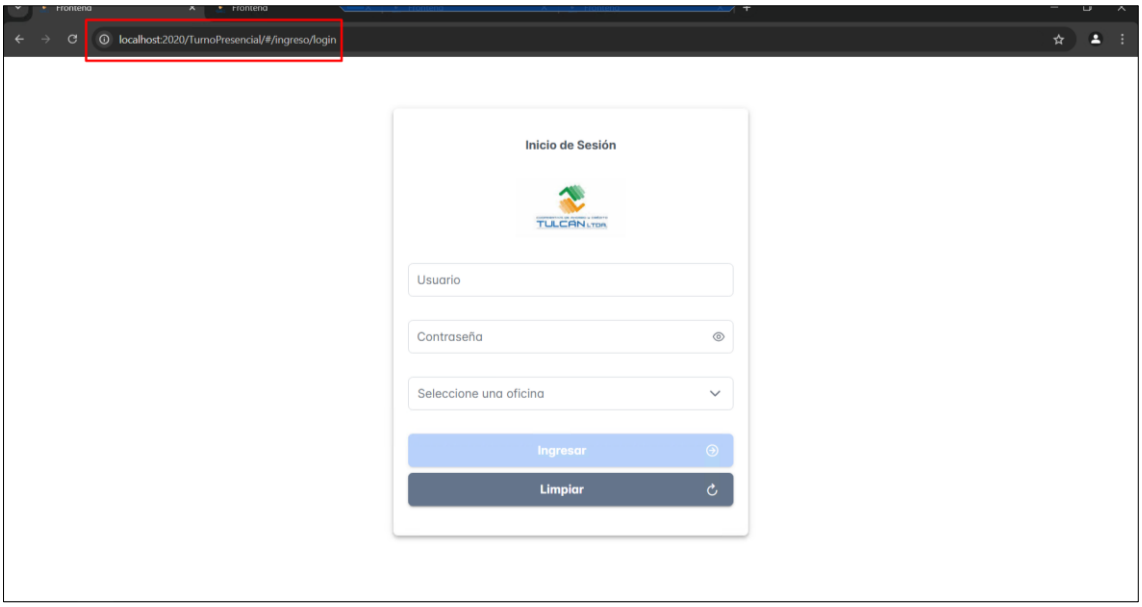


Reinicio de servidor de aplicaciones Apache

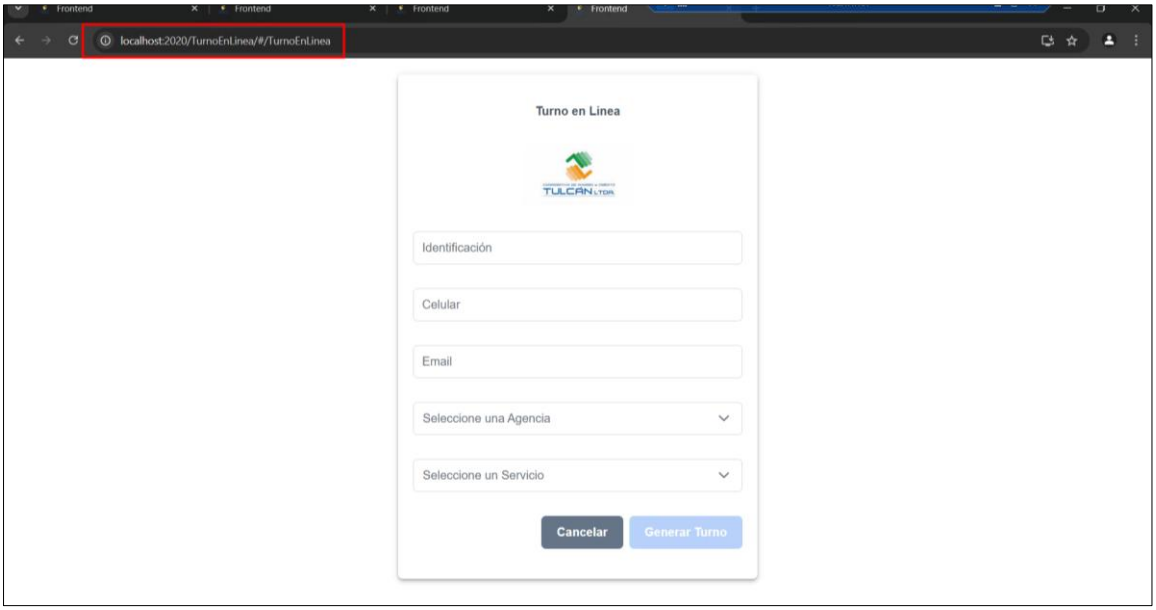


D 4. Pruebas y Verificación

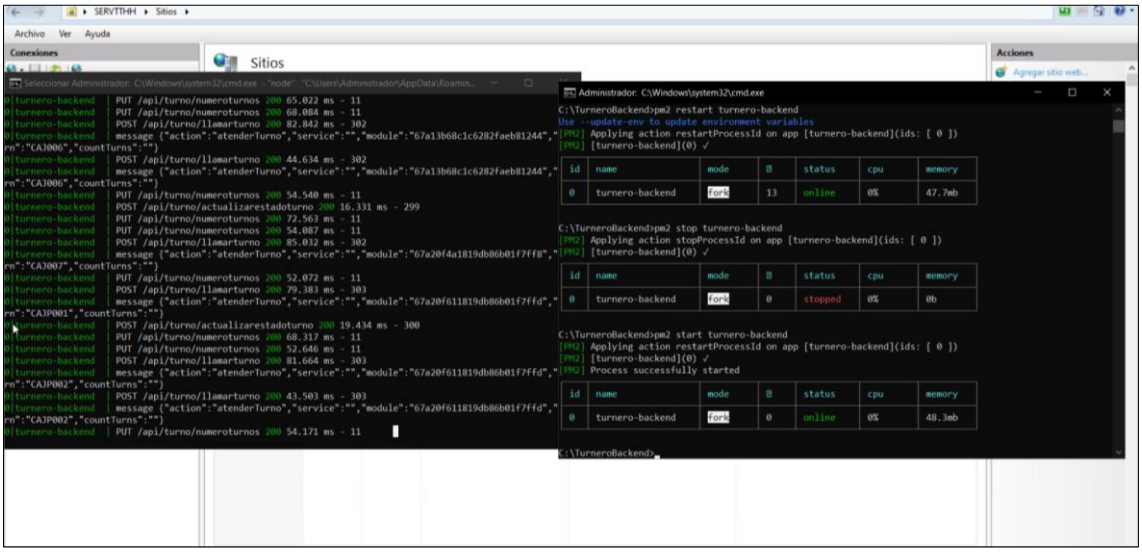
Validación de funcionamiento de turno presencial



Validación de funcionamiento de turno en línea



Monitoreo de logs



ANEXO E. Acta de autorización para la implementación del sistema

ACTA DE AUTORIZACIÓN PARA LA IMPLEMENTACIÓN DEL SISTEMA DE GESTIÓN DE TURNOS BASADO EN EL STACK MEAN

Lugar: Cooperativa de Ahorro y Crédito Tulcán Ltda., Matriz Tulcán
Fecha: 04 de junio de 2024

El presente documento tiene como objetivo Autorizar formalmente la implementación del "Sistema de Gestión de Turnos" desarrollado con el stack tecnológico MEAN (MongoDB, Express.js, Angular y Node.js), presentado como parte del proyecto de titulación "Marco tecnológico con el uso del Stack MEAN aplicado a la gestión de turnos en cooperativas de ahorro y crédito" previa verificación de su alineación con los requerimientos operativos y tecnológicos de la Cooperativa de Ahorro y Crédito Tulcán.

Aprobación del oficial de atención al cliente:

- Se reconoce la pertinencia del sistema para optimizar los procesos de asignación y seguimiento de turnos, mejorando la experiencia del usuario y la eficiencia operativa.
- Se valida que los requerimientos funcionales (RF) descritos en el documento técnico cubren las necesidades de las agencias.

Aprobación de tecnologías de la información:

- Se confirma la compatibilidad del stack MEAN con la infraestructura tecnológica existente.
- Se autoriza el uso de los siguientes recursos: Servidores para despliegue en ambiente de producción, servidores para despliegue en ambiente de desarrollo, acceso a información necesario

COOPERATIVA DE AHORRO Y CRÉDITO "TULCÁN" LTDA. ING. GABRIEL GORDON B. JEFE DE TECNOLOGÍA

Gabriel Romeo Gordon Benítez
CI: 0401126198
Jefe de Tecnología

TULCÁN LTDA. SUBGERENTE DE TI Y OPERACIONES

Yury Ravel Barahona Vasco
CI: 1707989032
Subgerente de Tecnología y Operaciones

COOPERATIVA FINANCIERA DE AHORRO Y CRÉDITO "TULCÁN" LTDA. ING. PAOLA MORALES OFICIAL ATENCIÓN AL CLIENTE

Adriana Paola Morales Cruz
CI: 0401441449
Oficial de atención al cliente

Calle Antisana y Av. Universitaria
Telf: (06) 2980837 - 2984435
info@upec.edu.ec
www.upec.edu.ec
Tulcán - Ecuador

ANEXO F. Acta de prueba de desarrollo del software/sistema

ACTA DE PRUEBA DE DESARROLLO DE SOFTWARE/SISTEMA DE INFORMACIÓN

Fecha	Día	Mes	Año
	11	11	2024

Nombre proyecto/sistema de Información	Sistema de gestión de turnos presencial y en línea

Se detallan las pruebas de funcionalidad realizadas sobre el sistema y se resumen las verificaciones, siendo todas estas exitosas

MÓDULO	DESCRIPCIÓN	RESULTADO	OBSERVACION
Módulo de inicio de sesión	Ingreso al sistema con un usuario y contraseña autogenerados y cambiar la contraseña	Éxito	Ninguna
Módulo de toma de turno presencial	El sistema permite tomar un turno en un servicio especifico	Éxito	Se debe incluir un teclado virtual para pantalla táctil
Módulo de toma de turno presencial	Se imprime térmicamente el código del turno y servicio con código QR de validación	Éxito	Ampliar el tamaño de fuente del código de turno
Módulo de toma de turno en línea	El sistema permite tomar un turno en una agencia y servicio específicos vía internet	Éxito	Ninguna
Módulo de visualización de turnos pendientes	Se puede visualizar los turnos pendientes y agendados en tiempo real	Éxito	Se debe poder el acceder sin inicio de sesión
Módulo de atención al cliente	El sistema permite realizar la llamada de un turno con una alerta visual y auditiva	Éxito	Ninguna
Módulo de atención al cliente	El sistema permite marcar un turno como atendido o no atendido	Éxito	Ninguna
Módulo de atención al cliente	Se puede visualizar turnos en espera en tiempo real y el tiempo de atención actual	Éxito	Se deben visualizar los turnos transferidos en tiempo real
Módulo de administración	Se permite realizar el mantenimiento de todos los submódulos del sistema	Éxito	Ninguna

COOPERATIVA DE AHORRO Y CREDITO TULCAN LTDA.

Yhuliana Gissel Revelo Lopez
CI: 0401745443
Asistente de negocios

COOPERATIVA DE AHORRO Y CREDITO TULCAN LTDA.
21 MAR 2025

Edwin Alexander Enriquez Lopez
CI: 0401251849
Asistente de Ventanilla

COOPERATIVA FINANCIERA DE AHORRO Y CREDITO "TULCAN LTDA."
ING. PAOLA MORALES
OFICIAL ATENCIÓN AL CLIENTE

Adriana Paola Morales Cruz
CI: 0401441449
Oficial de atención al cliente

ANEXO G. Acta de prueba de desarrollo software/sistema final

ACTA DE PRUEBA DE DESARROLLO DE SOFTWARE/SISTEMA DE INFORMACIÓN

Fecha	Día	Mes	Año
	15	01	2025

Nombre proyecto/sistema de Información	Sistema de gestión de turnos presencial y en línea

Se detallan las pruebas de funcionalidad realizadas sobre el sistema y se resumen las verificaciones, siendo todas estas exitosas

MÓDULO	DESCRIPCIÓN	RESULTADO	OBSERVACION
Módulo de inicio de sesión	Se implementa control de verificación de seguridad en contraseña nueva	Éxito	Ninguna
Módulo de toma de turno presencial	El sistema por medio de un parámetro solicita o no el ingreso de una identificación	Éxito	Ninguna
Módulo de toma de turno presencial	Se restringe ciertas operaciones en táctil para evitar cierre involuntario	Éxito	Ninguna
Módulo de toma de turno en línea	El sistema solicita completar un captcha después de 5 intentos fallidos de toma de turno en línea	Éxito	Ninguna
Módulo de visualización de turnos pendientes	La voz puede ser masculina o femenina parametrizable antes de la voz se emite un sonido de campana	Éxito	Ninguna
Módulo de visualización de turnos pendientes	Se aumenta tamaño de fuente y se cambia colores por los de la institución	Éxito	Ninguna
Módulo de atención al cliente	El sistema es accesible desde Raspberry Pi en monitores de visualización	Éxito	Ninguna
Módulo de atención al cliente	El sistema permite transferir un turno a otro módulo disponible en caso de ser requerido	Éxito	Ninguna
Módulo de administración	Se permite realizar el mantenimiento de todos los submódulos del sistema	Éxito	Ninguna

COOPERATIVA DE AHORRO Y CREDITO TULCÁN LTDA.

Yhuliana Gisset Revelo Lopez
CI: 0401745443
Asistente de negocios

COOPERATIVA DE AHORRO Y CREDITO TULCÁN LTDA.
21 MAR 2025

Edwin Alexander Enriquez Lopez
CI: 0401251889
Asistente de Ventanilla

COOPERATIVA FINANCIERA DE AHORRO Y CREDITO "TULCAN" LTDA.
ING. PAOLA MORALES
OFICIAL ATENCIÓN AL CLIENTE

Adriana Paola Morales Cruz
CI: 0401441449
Oficial de atención al cliente

ANEXO H. Acta de entrega de Software de Nivel Funcional

ACTA DE ENTREGA DE SOFTWARE DE NIVEL FUNCIONAL

Fecha	Día	Mes	Año
	11	02	2025

Nombre proyecto/sistema de Información	Sistema de gestión de turnos presencial y en línea

El presente documento tiene como objetivo formalizar la entrega del sistema de gestión de turnos, desarrollado por Iván Darío Rojas Rojas, para la Cooperativa de Ahorro y Crédito Tulcán. Este software cumple con todos los requisitos funcionales, especificaciones técnicas y requerimientos establecidos, garantizando su alineación con las necesidades y expectativas de la organización.

Detalles Técnicos de la entrega

Funcionalidades entregadas:

Módulo de inicio de sesión.
Módulo de toma de turno presencial.
Módulo de toma de turno en línea.
Módulo de visualización de turnos pendientes.
Módulo de atención al cliente.
Módulo de administración.

Documentación entregada:

Manual de usuario.
Guía rápida de inicio.

COOPERATIVA FINANCIERA DE AHORRO Y CRÉDITO "TULCÁN" LTDA.
ING. PAOLA MORALES
OFICIAL ATENCIÓN AL CLIENTE

Adriana Paola Morales Cruz
CI: 0401441449
Oficial de atención al cliente

COOPERATIVA DE AHORRO Y CRÉDITO "TULCÁN" LTDA.
ING. GABRIEL GORDÓN B.
JEFE DE TECNOLOGÍA

Gabriel Romeo Gordon Benitez
CI: 0401126198
Jefe de Tecnología

MARKETING

Pablo Eduardo Hernández Rosero
CI: 0401140033
Jefe de Marketing

Iván Darío Rojas Rojas
CI: 0401709035
Analista de Tecnología

Calle Antisana y Av. Universitaria
Telf: (06) 2980837 - 2984435
info@upec.edu.ec
www.upec.edu.ec
Tulcán - Ecuador

ANEXO I. Acta de entrega de Software de Nivel Técnico

ACTA DE ENTREGA DE SOFTWARE DE NIVEL TÉCNICO

Fecha	Día	Mes	Año
	04	02	2025

Nombre proyecto/sistema de Información	Sistema de gestión de turnos presencial y en línea

El presente documento tiene como objetivo formalizar la entrega técnica del sistema de gestión de turnos, desarrollado por Iván Darío Rojas Rojas, al equipo técnico encargado de su mantenimiento. Este software ha sido desarrollado cumpliendo con los requisitos funcionales, especificaciones técnicas establecidos, garantizando su correcto funcionamiento y adaptabilidad para su puesta en producción. La entrega incluye toda la documentación técnica necesaria, así como los recursos y componentes requeridos para facilitar su configuración y mantenimiento continuo.

Detalles Técnicos de la entrega

Tecnologías Utilizadas:

Lenguaje de programación: JavaScript, TypeScript
Base de datos: MongoDB
Frameworks: Angular y Express
Servidores: servidor HTTP integrado Node.js

Puesta en marcha del sistema:

Servidor de base de datos inicializado, backend iniciado de Node.js con ayuda de pm2 y servidor de aplicación frontend Apache iniciado.

Entregables Técnicos:

Código fuente.
Acceso al repositorio de control de versiones.
Scripts de base de datos.
Archivos de configuración.
Documentación técnica.

COOPERATIVA DE AHORRO Y CREDITO "TULCAN" LTDA.
ING. GABRIEL GORDÓN B.
JEFE DE TECNOLOGÍA

Gabriel Romeo Gordon Benítez
CI: 0401126198
Jefe de Tecnología

TULCAN LTDA.
SUBGERENTE DE TIC Y OPERACIONES

Yury Pavel Barahona Vasco
CI: 1707989032
Subgerente de Tecnología y Operaciones

Iván Darío Rojas Rojas
CI: 0401709035
Analista de Tecnología

Calle Antisana y Av. Universitaria
Telf: (06) 2980837 - 2984435
info@upec.edu.ec
www.upec.edu.ec
Tulcán - Ecuador