UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN

Tema:

APLICACIÓN MÓVIL PARA DISPOSITIVOS ANDROID APLICANDO MODELOS DE PREDICCIÓN PARA LA GESTIÓN DE PEDIDOS EN RAMONES RESTAURANT.

Trabajo de titulación modalidad Proyecto de Investigación, presentado previo a la obtención del título de Ingeniero en Tecnologías de la Información

ÁREA: Software

LÍNEA DE INVESTIGACIÓN: Tecnologías de la información y Sistemas de control

AUTOR: Cristian Javier López Pérez

TUTOR: Ing. Oscar Fernando Ibarra Torres, Mg.

Ambato - Ecuador

febrero – 2024

APROBACIÓN DEL TUTOR

En calidad de tutor del trabajo de titulación con el tema: APLICACIÓN MÓVIL PARA DISPOSITIVOS ANDROID APLICANDO MODELOS DE PREDICCIÓN PARA LA GESTIÓN DE PEDIDOS EN RAMONES RESTAURANT, desarrollado bajo la modalidad Proyecto de Investigación por el señor Cristian Javier López Pérez, estudiante de la Carrera de Tecnologías de la Información de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que el estudiante ha sido tutorado durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 17 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.3 del instructivo del reglamento referido.

Ambato, febrero 2024.

Firmado electrónicamente por:
OSCAR FERNANDO
IBARRA TORRES

Ing. Oscar Fernando Ibarra Torres, Mg.

TUTOR

AUTORÍA

El presente trabajo de titulación con el tema: APLICACIÓN MÓVIL PARA DISPOSITIVOS ANDROID APLICANDO MODELOS DE PREDICCIÓN PARA LA GESTIÓN DE PEDIDOS EN RAMONES RESTAURANT es absolutamente original, auténtico y personal y ha observado los preceptos establecidos en la Disposición General Quinta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato. En tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, febrero 2024.

Cristian Javier López Pérez

C.C. 2300372378

AUTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato para que reproduzca total o parcialmente este trabajo de titulación dentro de las regulaciones legales e institucionales correspondientes. Además, cedo todos mis derechos de autor a favor de la institución con el propósito de su difusión pública, por lo tanto, autorizo su publicación en el repositorio virtual institucional como un documento disponible para la lectura y uso con fines académicos e investigativos de acuerdo con la Disposición General Cuarta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato.

Ambato, febrero 2024.

Cristian Javier López Pérez

C.C. 2300372378

AUTOR

APROBACIÓN DEL TRIBUNAL DE GRADO

En calidad de par calificador del informe final del trabajo de titulación presentado por el señor Cristian Javier López Pérez, estudiante de la Carrera de Tecnologías de la Información de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Proyecto de Investigación, titulado APLICACIÓN MÓVIL PARA DISPOSITIVOS ANDROID APLICANDO MODELOS DE PREDICCIÓN PARA LA GESTIÓN DE PEDIDOS EN RAMONES RESTAURANT, nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 19 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.4 del instructivo del reglamento referido. Para cuya constancia suscribimos, conjuntamente con la señora Presidente del Tribunal.

Ambato, febrero 2024.

Firmado electrónicamente por:
ELSA PILAR URRUTIA URRUTIA

Ing. Elsa Pilar Urrutia Urrutia, Mg.
PRESIDENTE DEL TRIBUNAL

Firmado electrónicamente por:
DANIEL SEBASTIAN JEREZ MAYORGA

Ing. Mg. Daniel Jeréz Mayorga
PROFESOR CALIFICADOR

Firmado electrónicamente por:
JULIO ENRIQUE BALAREZO LOPEZ

Ing. PhD. Julio Balarezo López
PROFESOR CALIFICADOR

DEDICATORIA

Quiero dedicar este trabajo a mis amados padres, quienes han sido mi principal apoyo a lo largo de mi vida, siendo mis principales maestros, con el ejemplo, me han enseñado el valor del esfuerzo y la perseverancia. Este logro es también suyo, reflejo de su amor inagotable y su incansable apoyo. Les dedico cada éxito, como ustedes me han dedicado cada día de su amor y guía.

A mis queridos hermanos, compañeros de mil aventuras. Ustedes han añadido alegría y fortaleza a mi camino. En cada desafío y en cada triunfo, siempre he sentido su aliento y su orgullo fraterno. Este logro es un testimonio de los lazos indestructibles que nos unen.

A mi abuelita y a mi tía, cuya sabiduría y cariño han sido mi guía constante. Este logro es un reflejo de su amor y fortaleza, que siempre han iluminado mi camino. Les dedico este éxito con todo mi amor y gratitud.

AGRADECIMIENTO

Debo agradecer de todo corazón a quienes han sido partícipes de toda mi formación académica, a mis docentes, por ser quienes han compartido con mucho tacto y paciencia todos sus conocimientos conmigo.

A mis amigos y colegas Roger, Deybi, Stiven y Bryann, en quienes siempre pude contar cuando más los necesité, y con quienes compartí muy bonitas experiencias durante este camino.

A mi tutor, Ingeniero Fernando Ibarra, que con su guía y consejos pude culminar con este objetivo, además de aprender mucho de él durante el proceso.

Y en especial, a ti, mi amor, en ti encontré no solo una pareja, sino también la madre excepcional de nuestra hija, el regalo más grande y precioso de mi vida. Este éxito es también tuyo, un pequeño gesto para compensar todo lo que has dado y significas para mí y para nuestra pequeña. Te amo y te dedico cada logro, cada sueño alcanzado, como un reflejo del amor y la felicidad que tú y nuestra hija han traído a mi vida.

Esto es gracias a todos ustedes, y para ustedes, nada de esto sería posible de no ser por su imprescindible presencia en mi vida.

ÍNDICE GENERAL DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

ÍNDICE DE ANEXOS

RESUMEN EJECUTIVO

Este proyecto de investigación aborda el desarrollo e implementación de una aplicación móvil para dispositivos Android, utilizando modelos de predicción en la gestión de pedidos de Ramones Restaurant. El objetivo principal fue optimizar el proceso de atención al cliente mediante la integración de tecnología en el sistema tradicional de gestión de pedidos. Se realizó un análisis profundo de los procedimientos existentes en el restaurante, identificando áreas de mejora. Se evaluaron diversos modelos predictivos, enfocándose en su adaptabilidad, escalabilidad y potencial de integración en la estructura operativa del restaurante. Las redes neuronales fueron las adecuadas debido a su arquitectura flexible.

La aplicación se desarrolló utilizando diversos frameworks, lo que facilitó los distintos procesos de desarrollo. Se empleó la metodología ágil Scrumban para asegurar la adaptabilidad a los requisitos cambiantes y a los desafíos imprevistos. La implementación de la aplicación demostró la factibilidad y los beneficios de introducir tecnologías avanzadas en las PYMEs. Los hallazgos sugieren que dichas integraciones pueden lograrse mediante una planificación cuidadosa y la consideración de las complejidades específicas del negocio.

El estudio recomienda un análisis comparativo más profundo de los modelos predictivos para optimizar el rendimiento de la aplicación. Las investigaciones futuras deberían explorar el impacto más amplio de la implementación de tecnologías en las PYMEs. Además, se aconseja para Ramones Restaurant implementar un plan de evaluación post-implementación para asegurar mejoras continuas y la adaptación de la aplicación a los cambios del negocio y las expectativas de los clientes.

Palabras clave: Aplicación móvil, Android, modelos de predicción, gestión de pedidos, frameworks, Scrumban, PYMEs, adaptabilidad.

ABSTRACT

This research project focuses on the development and implementation of an Android mobile application using predictive models for order management at Ramones Restaurant. The primary objective was to optimize the company's customer service process by integrating technology into its traditional order management system. An in-depth analysis of the existing order management procedures at Ramones Restaurant was conducted, revealing significant inefficiencies and areas for improvement. Various predictive models were evaluated, with a focus on their adaptability, scalability, and potential for integration into the restaurant's operational structure. Artificial neural networks were identified as the most suitable due to their flexible architecture.

The mobile application was developed using robust software frameworks, facilitating a faster and more efficient development process. The mobile application was developed using the Scrumban agile methodology, to ensure adaptability to changing requirements and unforeseen challenges. The implementation of the application demonstrated the feasibility and benefits of introducing advanced technologies in small and medium-sized enterprises (SMEs). The findings suggest that such integrations can be successfully achieved with careful planning and consideration of business-specific complexities.

The study recommends further comparative analysis of predictive models to optimize the application's performance. Future research should explore the broader impact of technology implementation in SMEs. Additionally, a post-implementation evaluation plan for Ramones Restaurant is advised to ensure continuous improvement and adaptation of the application to meet evolving business needs and customer expectations.

Keywords: Mobile application, Android, predictive models, order management, frameworks, Scrumban, SMEs, adaptability.

CAPÍTULO I. MARCO TEÓRICO

1.1 Tema de investigación

APLICACIÓN MÓVIL PARA DISPOSITIVOS ANDROID APLICANDO MODELOS DE PREDICCIÓN PARA LA GESTIÓN DE PEDIDOS EN RAMONES RESTAURANT.

1.1.1 Planteamiento del problema

A lo largo del tiempo, la humanidad ha experimentado una evolución tecnológica que ha impactado en diversos aspectos de la vida cotidiana, incluyendo la implementación de Tecnologías de la Información (TI) en las organizaciones.

En el contexto de las pequeñas y medianas empresas (pymes) en Ecuador, se ha observado un creciente interés por adoptar recursos informáticos, especialmente debido a los cambios en el comportamiento consumista impulsados por la pandemia. La implantación de las TI en las pymes tiene como objetivo lograr una transformación digital que mejore la productividad y eficiencia de la empresa, adaptando el modelo de negocio a las nuevas tendencias del mercado.[1]. Sin embargo, este proceso puede enfrentar resistencia al cambio y obstáculos relacionados con la falta de flexibilidad en las organizaciones tradicionales.[2].

En el centro del país, una región con una presencia industrial destacada, tanto empresas de gran escala como pymes dedicadas a la comida y bebida han adoptado sistemas informáticos para optimizar procesos, recolectar información y mejorar la atención al cliente.[3]. En este contexto, Ramones Restaurant, una empresa en crecimiento dentro de la industria alimentaria, busca implementar tecnologías que mejoren la experiencia del cliente, agilicen los procesos internos, generen resultados rentables y respalden la toma de decisiones. La adopción de herramientas y sistemas informáticos específicos permitirá recopilar datos en tiempo real, identificar áreas de mejora, optimizar el rendimiento general del restaurante y mejorar la eficiencia operativa. Esto se traducirá en una experiencia positiva para los clientes, mayor satisfacción y fidelidad hacia el restaurante, así como en un aumento de la rentabilidad del negocio.

1.2 Antecedentes investigativos

Para poder tener una mayor comprensión del presente proyecto, se llevó a cabo una extensa recopilación de distintas fuentes bibliográficas referentes al tema planteado, que fungen como base de conocimiento para el desarrollo del proyecto.

Pazo S. y Toaquiza P. [4] explican como su principal objetivo es desarrollar e implementar un aplicativo móvil para monitoreo, control de las solicitudes y entregas de pedidos mediante geolocalización en tiempo real en dicha empresa con la finalidad de mejorar los servicios y la comunicación con los clientes bajo la creación de esta aplicación, aplicando una metodología denominada Mobile-D, la cual se divide en varias fases con subprocesos iterativos, y se ajusta a las necesidades establecidas para el desarrollo del proyecto definido por los autores. Concluyeron que el uso de Frameworks facilitan mucho la tarea de desarrollo porque al contar con un conjunto de herramientas predefinidas, se optimiza el tiempo y uso de recursos. Además, la metodología ágil Mobile-D resultó mejor debido a que se aplica a grupos de desarrollo reducidos.

El proyecto de investigación realizado por Zambrano M. [5] explica que su trabajo de investigación se realizó con el objetivo de mejorar las ventas de la empresa "GT Jean's Cupido" mediante la implementación de una plataforma tecnológica, utilizando el Framework Angular 8 y aplicando la metodología Xtreme Programming (XP), llegando a la conclusión que la identificación eficiente de los procesos de ventas y el análisis de la herramienta PWA (Progressive Web App) fueron fundamentales. La implementación de la tienda virtual web PWA permitió una gestión satisfactoria de las ventas en línea y brindó una experiencia positiva tanto para la empresa como para los clientes.

El trabajo de investigación realizado por Villanueva M. [6] se centra en el desarrollo de una aplicación móvil Android para la empresa Transportes Sanmartí SA, con el objetivo de gestionar en tiempo real los pedidos asignados a los transportistas. El proyecto se dividió en cinco fases y se implementaron funcionalidades clave, como seguimiento de pedidos, flujo de datos bidireccional y consultas de ubicación. Tras concluir el desarrollo, se evaluaron los resultados y se constató que se cumplieron

todos los requisitos y objetivos establecidos. Destaca la participación de la supervisora, el director de la empresa y el feedback del tutor, quienes contribuyeron al cumplimiento de los objetivos del proyecto. Concluyendo, se logró desarrollar una aplicación móvil que satisface las necesidades de gestión de pedidos en tiempo real de Transportes Sanmartí SA.

En un proyecto de desarrollo relacionado a la calidad del servicio al cliente, realizado por Martínez E. [7] manifiesta la relación entre la calidad en el servicio y la satisfacción del cliente en restaurantes de carnes al grill en la ciudad de Ambato. Se utilizaron dos modelos de medición, SERVQUALing y ACSI, y se aplicaron encuestas con 27 ítems relacionados a las dimensiones de ambos modelos, denotando una relación entre la calidad en el servicio y la satisfacción del cliente. Se identificaron áreas de mejora para satisfacer a los clientes de cada restaurante. Destaca la importancia de utilizar diferentes modelos de medición para obtener una visión más completa de la calidad en el servicio y la satisfacción del cliente. En conclusión, este estudio respalda la relación entre la calidad en el servicio y la satisfacción del cliente en los restaurantes de carnes al grill en Ambato. Estos resultados permiten a los propietarios establecer estrategias de fidelización y lealtad de los clientes, teniendo en cuenta las percepciones y necesidades actuales. Se destaca la relevancia de adaptar las encuestas a los modelos de medición utilizados y considerar la influencia del precio en relación con el servicio ofrecido.

Un proyecto similar utilizó la metodología XP en cuatro fases: planificación, diseño, codificación y prueba. La aplicación cuenta con módulos de compra, reservación y entrega de pedidos, y ofrece opciones de pago mediante transferencias bancarias y en efectivo. Se siguió una arquitectura cliente-servidor y modelo vista-controlador. Las pruebas de funcionalidad validaron el correcto desempeño de la aplicación, aunque se identificaron algunos desafíos durante la implementación de los módulos. En conclusión, se logró desarrollar una aplicación móvil que mejora la oferta y ventas del Supermarket Supercito. [8].

1.3 Fundamentación teórica

1.3.1 Gestión empresarial:

La Gestión Empresarial se refiere al conjunto de actividades y procesos destinados a administrar y dirigir una organización de manera eficiente y efectiva para lograr sus objetivos. Esto implica la planificación, organización, dirección y control de los recursos y actividades de la empresa, así como la toma de decisiones estratégicas para garantizar su éxito y crecimiento.[9].

1.3.2 Sistemas de gestión:

Los Sistemas de Gestión son marcos de trabajo y metodologías utilizados para gestionar diferentes aspectos de una organización de manera sistemática y estructurada. Estos sistemas abarcan áreas como la calidad, el medio ambiente, la seguridad laboral y la gestión de riesgos. Proporcionan pautas y procesos para establecer políticas, objetivos, procedimientos y controles, con el fin de mejorar el desempeño y la eficiencia de la organización.[10].

1.3.3 Atención al cliente y gestión de incidencias:

La Atención al Cliente se refiere a las acciones y procesos destinados a satisfacer las necesidades y expectativas de los clientes. Esto implica brindar un servicio de calidad, resolver consultas y problemas de manera efectiva, y mantener una comunicación fluida con los clientes. La Gestión de Incidencias se relaciona con la identificación, seguimiento y resolución de problemas o situaciones inesperadas que pueden surgir en la interacción con los clientes.[11].

1.3.4 Gestión de pedidos:

La Gestión de Pedidos se ocupa de todo el proceso relacionado con la recepción, procesamiento y entrega de pedidos realizados por los clientes. Esto incluye la captura de pedidos, la asignación de recursos necesarios, la planificación de la producción o distribución, el seguimiento del estado de los pedidos y la gestión de devoluciones. Una eficiente gestión de pedidos contribuye a mejorar la satisfacción del cliente, reducir los tiempos de entrega y optimizar los recursos de la empresa.[12].

a. Actividades para la gestión de restaurantes

Una organización tiene actividades que cumplir para poder existir, independientemente de la naturaleza de la empresa, tiene una estructura detallada del personal y los roles que desempeñan, y en base a esto es que cada parte que conforma a la organización sabe sus tareas y responsabilidades.[13].

En el marco de una empresa restaurantera, las actividades que se llevan a cabo suelen ser tradicionales a nivel general, debido a que se trata de un modelo de negocio existente desde inicios de la civilización, lo que lo dota de conocimiento empírico de que ofrece, como compite y adquiere valor en el mercado.[14].

En este caso, el enfoque está dado hacia la optimización en la gestión de los pedidos, por lo que se requiere desglosar las tareas y pasos que conlleva esto, desde el ingreso del cliente hasta su salida. Evidentemente, existen otras áreas en las que se realizan actividades para el óptimo funcionamiento de la empresa, sin embargo, no son objeto de estudio para el análisis.

La gestión de pedidos en un restaurante es un componente esencial en la prestación de un servicio de calidad. Este conjunto de tareas comprende una serie de actividades meticulosamente diseñadas para asegurar la satisfacción del cliente, las que comúnmente son:

- ***Toma del pedido***

El personal de servicio interactúa directamente con el cliente para tomar su pedido, utilizando técnicas de comunicación efectivas para asegurar la comprensión precisa de

las preferencias y necesidades del cliente, una vez confirmado el pedido, se documenta manualmente, detallando todos los elementos solicitados.

- ***Entrega del pedido a cocina***

El pedido se comunica al equipo de cocina, asegurando que la información sea clara y precisa para evitar errores en la preparación.

- ***Monitoreo del Estado del Pedido***

Se realiza un seguimiento constante del progreso de la preparación del pedido, para proporcionar estimaciones de tiempo realistas al cliente y gestionar sus expectativas.

- ***Entrega del Pedido al cliente***

Al estar listo, el pedido se entrega al cliente con un estándar alto de presentación, atendiendo a cualquier requerimiento adicional que pueda surgir en el momento.

Estas son las etapas fundamentales en la gestión de pedidos, pero se debe tener en cuenta que no son pasos estrictos, son susceptibles a cambios y ajustes, dependiendo de cómo oriente el negocio la atención al cliente.

1.3.5 Aplicaciones Móviles:

Las Aplicaciones Móviles se refieren a programas diseñados para ser ejecutados en dispositivos móviles, como smartphones y tablets. Estas aplicaciones aprovechan las características y capacidades de estos dispositivos, como la conectividad a Internet, los sensores integrados y las interfaces táctiles, para brindar servicios y funcionalidades específicas a los usuarios.[15].

1.3.6 Gestión de datos:

La Gestión de Datos se ocupa de organizar, almacenar, procesar y administrar la información de manera eficiente y segura. Esto implica la recolección, el almacenamiento y el análisis de datos para generar conocimientos y apoyar la toma de

decisiones. La Gestión de Datos se basa en técnicas y herramientas que permiten gestionar grandes volúmenes de información de manera estructurada y confiable.[16].

1.3.7 Metodologías de desarrollo:

Una metodología es una colección de procedimientos, técnicas, herramientas y documentos auxiliares que ayudan a los desarrolladores de software en sus esfuerzos por implementar nuevos sistemas de información. Una metodología está formada por fases, cada una de las cuales se puede dividir en sub-fases, que guiarán a los desarrolladores de sistemas a elegir las técnicas más apropiadas en cada momento del proyecto y también a planificarlo, gestionarlo, controlarlo y evaluarlo.[17].

1.3.8 Aplicación móvil para Android aplicando modelos de predicción:

Se trata del desarrollo de una aplicación móvil específica para dispositivos Android que utiliza modelos de predicción como refuerzo para la toma de decisiones. Al aplicar Modelos de Predicción, la aplicación puede analizar datos históricos y variables relevantes para predecir la demanda de productos, el tiempo de entrega estimado, las preferencias de los clientes, entre otros aspectos, lo que facilita una gestión más eficiente y precisa de los requerimientos del cliente.[18].

a. Aprendizaje automático

Los modelos de predicción son herramientas matemáticas que se utilizan para anticipar resultados futuros basados en datos históricos o condiciones actuales. Estos modelos van desde algoritmos sencillos, como una regresión lineal que predice una variable en función de otra, hasta modelos más complejos, como las redes neuronales artificiales que procesan grandes cantidades de datos para realizar predicciones más precisas. Son aplicables a distintos campos y su principal función es asistir en la toma de decisiones y optimización de procesos.

En el aprendizaje automático, los modelos de predicción se clasifican generalmente en tres categorías principales: aprendizaje supervisado, no supervisado y por refuerzo.

- ***Aprendizaje Supervisado***

Este tipo de aprendizaje utiliza datos etiquetados para entrenar modelos. El modelo aprende de los datos de entrada y las salidas correspondientes para hacer predicciones o clasificaciones en nuevos datos. Ejemplos comunes son la regresión lineal y las redes neuronales para clasificación.

- ***Aprendizaje No Supervisado***

A diferencia del supervisado, el aprendizaje no supervisado trabaja con datos no etiquetados. Se enfoca en identificar patrones o estructuras intrínsecas en los datos. La agrupación (clustering) y la reducción de dimensionalidad son ejemplos típicos de este tipo.

- ***Aprendizaje por Refuerzo***

En este enfoque, un agente aprende a tomar decisiones mediante la experimentación en un entorno. Se basa en el concepto de recompensas y penalizaciones para enseñar al modelo qué acciones son preferibles. Se utiliza comúnmente en campos como juegos y robótica.

b. Algoritmos de aprendizaje automático

- ***Regresión Lineal***

La regresión lineal es un modelo estadístico que busca establecer una relación lineal entre una variable dependiente y una o más variables independientes. Es un modelo fácil de implementar y de entender, lo que facilita su interpretación en aplicaciones empresariales. Orientado a la gestión de pedidos, la regresión lineal podría usarse para predecir tiempos de entrega en función de variables como la afluencia de la clientela, el volumen del pedido, la hora del día, entre otras, permitiendo así una planificación más eficiente.[19].

- ***Árboles de Decisión***

Los árboles de decisión son estructuras jerárquicas que utilizan un enfoque de "si-entonces" para tomar decisiones. El modelo divide el conjunto de datos en

subconjuntos más pequeños mientras desarrolla de manera asociada un árbol de decisión incremental. Son especialmente útiles para tareas de clasificación y pueden manejar tanto datos categóricos como numéricos. Podrían ayudar a clasificar la prioridad de los pedidos en función de varios factores como el tipo de producto, la cantidad y el tiempo de llegada del cliente, optimizando así el proceso de atención.

- ***Máquinas de Soporte Vectorial (SVM)***

Las Máquinas de Soporte Vectorial son modelos que buscan encontrar el hiperplano que mejor separa los datos en clases diferentes. Este modelo es eficiente en espacios de alta dimensión y puede ser ajustado para problemas de regresión o de clasificación. En la gestión de pedidos, SVM podría usarse para asignar recursos de manera más eficiente, por ejemplo, determinando qué empleado es el más adecuado para manejar un tipo específico de pedido, mejorando así la eficiencia y la satisfacción del cliente.

- ***Redes Neuronales Artificiales***

Las redes neuronales artificiales son sistemas que intentan emular el comportamiento del cerebro humano. Pueden aprender de grandes cantidades de datos y son excepcionales en la identificación de patrones complejos. Dada su flexibilidad y poder, serían ideales para anticipar comportamientos de demanda muy variados y podrían ayudar en la gestión automática de inventarios y en la asignación dinámica de recursos humanos para la atención de pedidos.[20].

- ***Series Temporales (ARIMA, Prophet, etc.)***

Los modelos de series temporales como ARIMA o Prophet están diseñados para analizar y predecir datos que tienen una estructura temporal. Estos modelos son muy útiles para prever la demanda de pedidos en períodos de tiempo específicos, como días festivos o fines de semana. Esto permite a la empresa prepararse mejor para picos de demanda, optimizando tanto el inventario como la asignación de personal.

- ***Clustering (K-Means, DBSCAN, etc.)***

Los modelos de clustering como K-Means o DBSCAN se utilizan para agrupar objetos en conjuntos basados en la similitud de sus características. Estos modelos no necesitan etiquetas para los datos y son útiles para descubrir patrones intrínsecos. En la gestión de pedidos, el clustering podría utilizarse para segmentar a los clientes o los tipos de pedidos, lo que permitiría una personalización más eficaz de los servicios y una asignación más precisa de los recursos.

1.4 Objetivos

1.4.1 Objetivo general

Implementar una aplicación móvil para dispositivos Android aplicando modelos de predicción para la gestión de pedidos en "Ramones Restaurant".

1.4.2 Objetivos específicos

- Analizar el proceso de gestión de pedidos que lleva a cabo la Empresa.
- Determinar qué modelo de predicción se ajusta mejor a las actividades que realiza la Empresa.
- Desarrollar una aplicación móvil para dispositivos Android utilizando modelos de predicción para optimizar el proceso de atención al cliente en la Empresa.

CAPÍTULO II. METODOLOGÍA

2.1 Materiales

Para poder elaborar este capítulo, se requirió de la elaboración y aplicación de la técnica de recolección de información que se ajustó a la naturaleza de la investigación, de la mano de su respectivo instrumento, el cual fue el medio principal para poder llegar a obtener resultados.

Se diseñó una encuesta que va dirigida al personal de la Empresa; meseros, cocineros y administrativos, la cual tuvo como objetivo analizar la percepción del personal acerca de la gestión actual que existe en el negocio, apreciar la satisfacción que tienen con el sistema de comandas y que tan bueno les resulta para sus tareas, para que de esta manera se evalúe que tan factible resulta implementar una solución tecnológica, en este caso, una aplicación móvil para optimizar la gestión de pedidos.

ESTUDIO DE PERCEPCIÓN Y FACTIBILIDAD DE LA APLICACIÓN MÓVIL PARA LA GESTIÓN DE PEDIDOS EN RAMONES RESTAURANT

Dirigido a *Encuesta dirigida al personal que conforma la empresa "Ramones Restaurant".*	**Encuestador** *Cristian López*
Objetivo *Analizar la percepción del personal sobre la gestión de pedidos en Ramones Restaurant y evaluar su satisfacción.*	
Indicaciones *La presente encuesta tiene preguntas con varios literales, escoja uno por cada pregunta según su criterio, subrayando la respuesta en cuestión.*	

1. ¿Ha experimentado dificultades en la coordinación y comunicación del proceso de pedidos en su trabajo actualmente?
a. Siempre
b. A menudo
c. A veces
d. Rara vez
e. Nunca

2. ¿Con qué frecuencia experimenta retrasos en los pedidos en Ramones Restaurant actualmente?
a. Siempre
b. A menudo
c. A veces
d. Rara vez
e. Nunca

3. ¿Está satisfecho/a con la rapidez actual en la entrega de pedidos en Ramones Restaurant?
a. Muy satisfecho/a
b. Satisfecho/a
c. Neutral
d. Insatisfecho/a
e. Muy insatisfecho/a

4. ¿Experimenta dificultades para acceder a la información relevante de los pedidos en sus labores diarias?
a. Siempre
b. A menudo
c. A veces
d. Rara vez
e. Nunca

5. ¿En qué áreas de la gestión de pedidos cree que podrían hacerse mejoras en Ramones Restaurant?
a. Tomar pedidos
b. Preparación de pedidos
c. Entrega de pedidos
d. Coordinación interna

6. ¿Cree que la gestión de pedidos puede mejorarse para ofrecer un servicio más rápido y efectivo a los clientes?
a. Sí
b. No

7. ¿Cómo calificaría la eficiencia actual en la gestión de pedidos en Ramones Restaurant?
a. Muy eficiente
b. Eficiente
c. Neutral
d. Poco eficiente
e. Ineficiente

8. ¿Considera que una mejor gestión de pedidos podría contribuir a la satisfacción del cliente?
a. Sí
b. No

9. ¿Qué tipo de dispositivo electrónico le resulta más fácil manipular o manejar?
a. Tablet
b. Computador de escritorio
c. Teléfono
d. Portátil

10. ¿Ha utilizado aplicaciones móviles en dispositivos Android previamente?
a. Sí
b. No

11. ¿Cree que una aplicación móvil facilitaría la gestión de pedidos y mejoraría su desempeño laboral?
a. Sí
b. No

12. ¿Considera que una aplicación móvil para gestionar pedidos podría reducir los errores en la toma y preparación de pedidos?
a. Sí
b. No

13. ¿Qué tan dispuesto/a estaría a utilizar una aplicación móvil para gestionar pedidos en su trabajo?
a. Altamente dispuesto/a
b. Muy dispuesto/a
c. Dispuesto/a
d. Poco dispuesto/a
e. Nada dispuesto/a

14. ¿Le gustaría que se implantara una solución tecnológica para gestionar de manera eficiente los pedidos en Ramones Restaurant?
a. Sí
b. No

2.2 Métodos

2.2.1 Modalidad de la investigación

En la presente investigación se contemplaron los siguientes enfoques:

Investigación Bibliográfica

Esta modalidad se contempló dentro de la investigación ya que se tuvo que recabar información de distintas fuentes bibliográficas para poder abordar el problema de una mejor manera, además de poder analizar diferentes enfoques de cada autor y el cómo se llegó a plantear una solución en cada caso, dónde la naturaleza del problema tiende a cambiar.

Investigación Aplicada

Al estar tratando un problema de investigación, se tuvo como finalidad proponer una solución, y para el caso de este proyecto se planteó el desarrollo de un software que se ajusta a las necesidades y requerimientos del negocio, el cual responde ante la problemática principal que existe en la empresa.

2.2.2 Población y muestra

El personal en general de la organización representó la totalidad de la población para esta investigación, ya que son los principales individuos que participan en la actividad de análisis para este trabajo, la cual es la gestión de pedidos, formando parte de este proceso los meseros, los encargados de cocina y la caja, así como la administración principal de la organización, definidos de la siguiente manera. (Ver Tabla 1).

Tabla 1. Población de estudio

Población	Cantidad	Porcentaje
Gerente	1	5%
Personal encargado de caja	1	5%
Personal encargado de atención al cliente	8	42%
Personal encargado de la cocina	9	48%
Total	19	100%

Para este caso, no se requirió de la elaboración de una muestra, ya que la población no sobrepasa de 100 individuos, así que se trabajó con la totalidad de la población.

2.2.3 Recolección de información

La aplicación de la encuesta se realizó a través de la herramienta de Google Forms, la cual permite diseñar el instrumento y a su vez poder ser compartido en línea con la población objetivo, para recopilar los resultados.

Validación del instrumento

Para garantizar la fiabilidad y consistencia de los resultados que se obtuvieron de la aplicación de la encuesta, se aplicó la técnica Alfa de Cronbach, del que se obtiene un coeficiente que, a través de una escala dada, sirve para medir la fiabilidad del instrumento, y a grandes rasgos, saber si este no está sesgado en alguno de sus ítems.

Tabla 2. Niveles de confianza – Alfa de Cronbach

Rango	Nivel de Confianza
0,9 – 1	Excelente
0,8 – 0,9	Buena
0,7 – 0,8	Aceptable
0,6 – 0,7	Cuestionable
0,5 – 0,6	Pobre
Menor a 0,5	Inaceptable

El cálculo de este coeficiente requiere de tres variables, expresados en la Ecuación 1.

$$\boldsymbol{\alpha = \frac{K}{K-1} * \left[1 - \frac{\sum s^2}{S^2}\right]} \qquad (1)$$

Dónde:

K = Cantidad de ítems del instrumento

$\sum s^2$ = Sumatoria de las varianzas de los ítems

S^2 = Varianza total del instrumento

Aplicando la Ecuación 1 con los valores obtenidos del instrumento, se obtiene que:

$$\boldsymbol{\alpha = \frac{14}{14-1} * \left[1 - \frac{12,34}{61,47}\right]} \qquad (2)$$

$$\alpha = 0{,}86$$

Acorde a la Tabla 2, el coeficiente obtenido de 0,86 sitúa al instrumento en un nivel de confianza Bueno.

Resultados de la encuesta

Pregunta 1: ¿Ha experimentado dificultades en la coordinación y comunicación del proceso de pedidos en su trabajo actualmente?

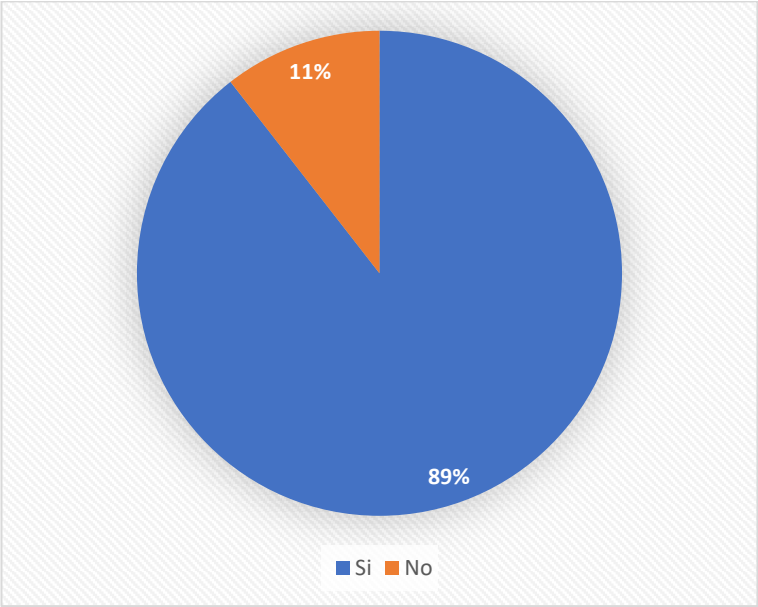


Figura 1. Tabulación de resultados pregunta 1

Análisis e Interpretación

Los resultados de la pregunta 1 demuestran que el 89% de la población ha experimentado alguna vez alguna dificultad ya sea en la coordinación o comunicación en el proceso de pedidos, mientras que el 11% no ha tenido este tipo de inconvenientes. Esto da indicio que la mayoría de la población es consciente de cuellos de botella en la gestión actual.

Pregunta 2: ¿Con qué frecuencia experimenta retrasos en los pedidos en Ramones Restaurant actualmente?

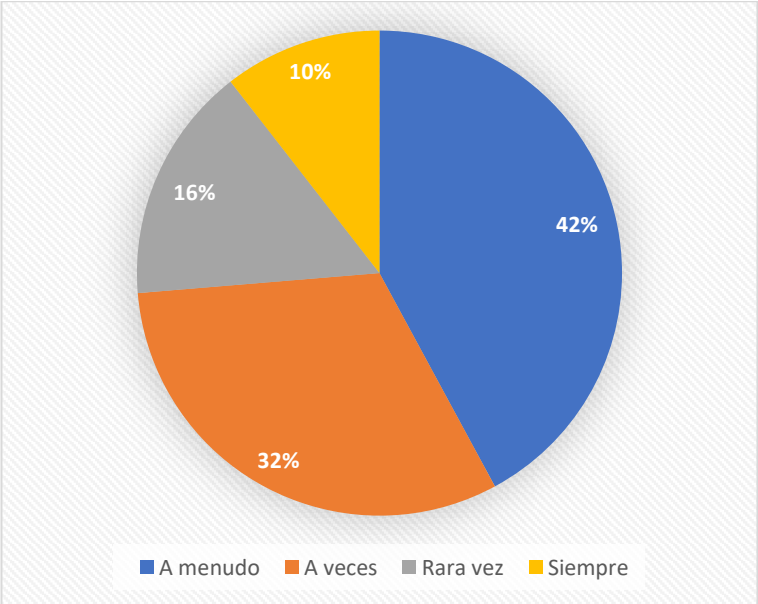


Figura 2. Tabulación de resultados pregunta 2

Análisis e Interpretación

Los resultados de la pregunta 2 indican que el 42% de los encuestados experimenta retrasos en los pedidos a menudo. Por otro lado, el 32% lo experimenta a veces, el 16% rara vez, y el 10% lo enfrenta siempre. Estos datos indican que la mayoría del personal de Ramones Restaurant experimentan retrasos de manera regular, a nivel general, en los pedidos.

Pregunta 3: ¿Está satisfecho/a con la rapidez actual en la entrega de pedidos en Ramones Restaurant?

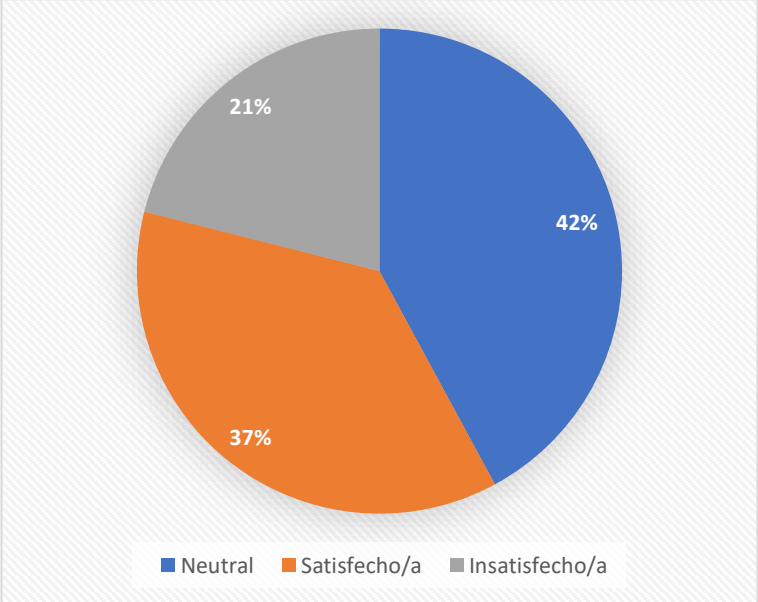


Figura 3. Tabulación de resultados pregunta 3

Análisis e Interpretación

Los resultados de la pregunta 3 demuestran que el 37% de los encuestados se siente satisfecho con la rapidez de entrega, mientras que el 42% tiene una opinión neutral. Además, el 21% se siente insatisfecho. Estos datos indican que, aunque una buena parte del personal se siente satisfecho con la rapidez con la que actualmente se cuenta para entregar los pedidos, existe un porcentaje significativo que no está del todo conforme con la velocidad de entrega en Ramones Restaurant.

Pregunta 4: ¿Experimenta dificultades para acceder a la información relevante de los pedidos en sus labores diarias?

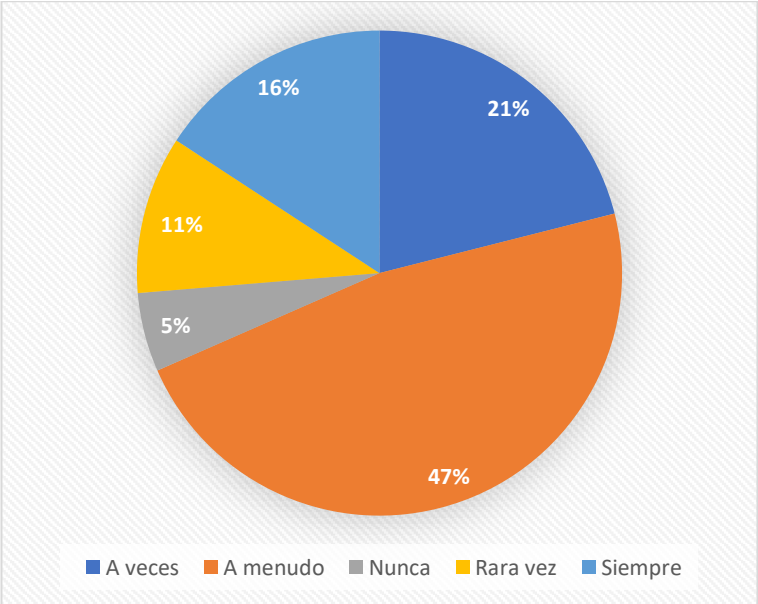


Figura 4. Tabulación de resultados pregunta 4

Análisis e Interpretación

Los resultados de la pregunta 4 abordan la problemática que la población enfrenta para poder mantenerse al tanto de cualquier información relevante para brindar una mejor atención al cliente, en lo que respecta a su pedido, un notable 47% experimenta problemas a menudo y un 16% afirma tener dificultades siempre. Esto indica que más del 60% de los encuestados enfrenta obstáculos recurrentes al acceder a la información de pedidos. Por otro lado, existe un 21% quienes a veces sufren estos inconvenientes, un 11% rara vez tiene problemas y solo un 5% nunca los experimenta.

Pregunta 5: ¿En qué áreas de la gestión de pedidos cree que podrían hacerse mejoras en Ramones Restaurant?

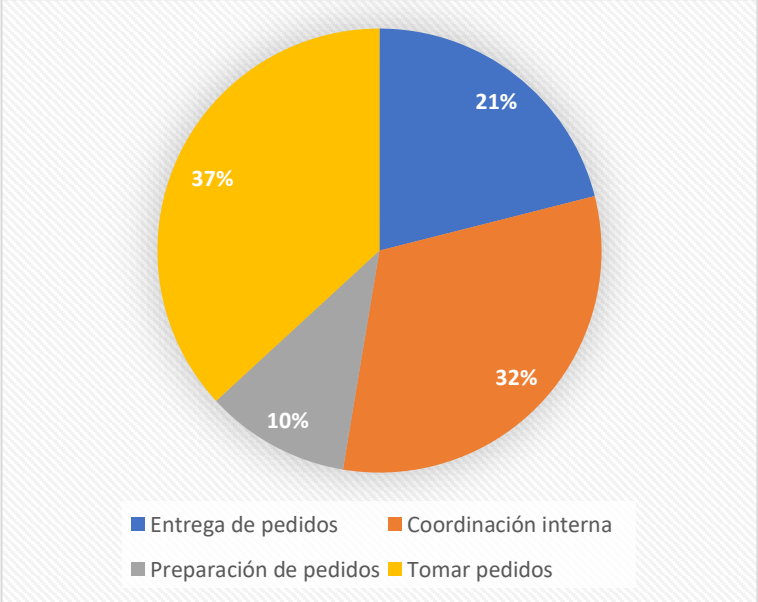


Figura 5. Tabulación de resultados pregunta 5

Análisis e Interpretación

Los resultados de la pregunta 5 dejan en evidencia que un 37% de las respuestas sugiere mejorar la toma de pedidos en Ramones Restaurant, mientras que un 32% ve problemas en la coordinación interna. La entrega y preparación de pedidos también fueron señaladas, con un 21% y 10% respectivamente. Estos datos indican que las áreas de inicio y finalización del proceso de pedido, junto con la coordinación interna, son las más críticas y podrían ser focos clave para optimizar en el restaurante.

Pregunta 6: ¿Cree que la gestión de pedidos puede mejorarse para ofrecer un servicio más rápido y efectivo a los clientes?

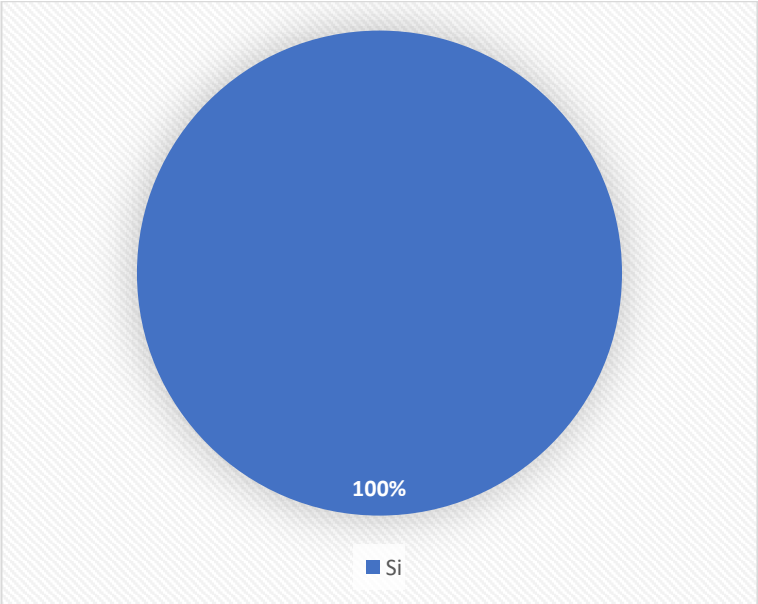


Figura 6. Tabulación de resultados pregunta 6

Análisis e Interpretación

Los resultados de la pregunta 6 son contundentes, el 100% de la población cree que, a nivel general, pueden realizarse mejoras en la gestión actual.

Pregunta 7: ¿Cómo calificaría la eficiencia actual en la gestión de pedidos en Ramones Restaurant?

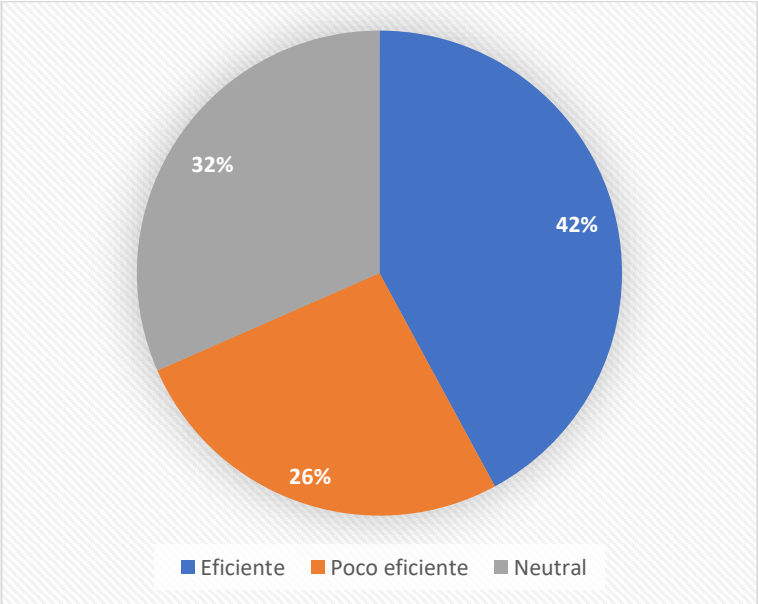


Figura 7. Tabulación de resultados pregunta 7

Análisis e Interpretación

El 42% de los encuestados considera que la gestión de pedidos en Ramones Restaurant es "Eficiente". Sin embargo, un 32% se mantiene "Neutral" y otro 26 % la ve como "Poco eficiente". Aunque la percepción general se inclina hacia una gestión eficiente, hay espacio para mejoras, dado el porcentaje significativo que señala ineficiencia o neutralidad.

Pregunta 8: ¿Considera que una mejor gestión de pedidos podría contribuir a la satisfacción del cliente?

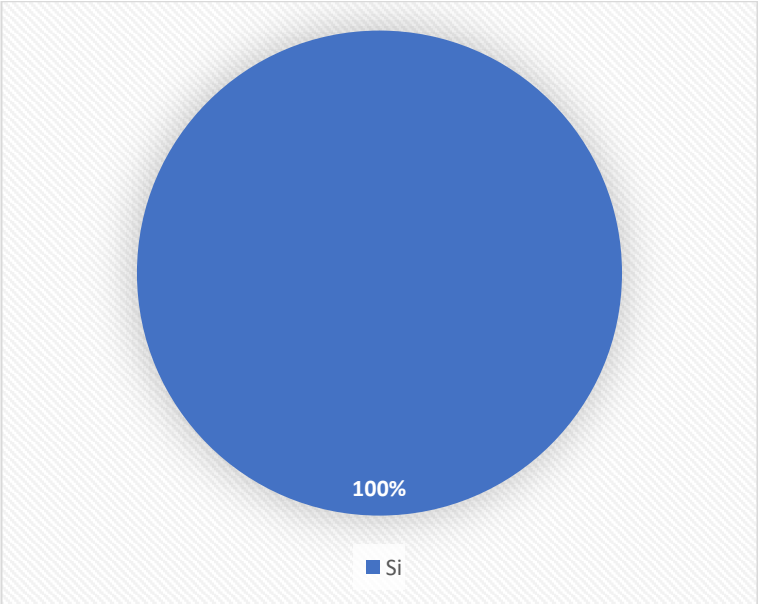


Figura 8. Tabulación de resultados pregunta 8

Análisis e Interpretación

Los resultados de la pregunta 8 son claros, las mejoras en el proceso actual están pensadas en pro de la atención y satisfacción del cliente, por lo que el 100% de la población concuerda en que, mejorando uno o varios aspectos de la gestión actual, la aceptación y comentarios de los clientes sobre la empresa mejorarán.

Pregunta 9: ¿Qué tipo de dispositivo electrónico le resulta más fácil manipular o manejar?

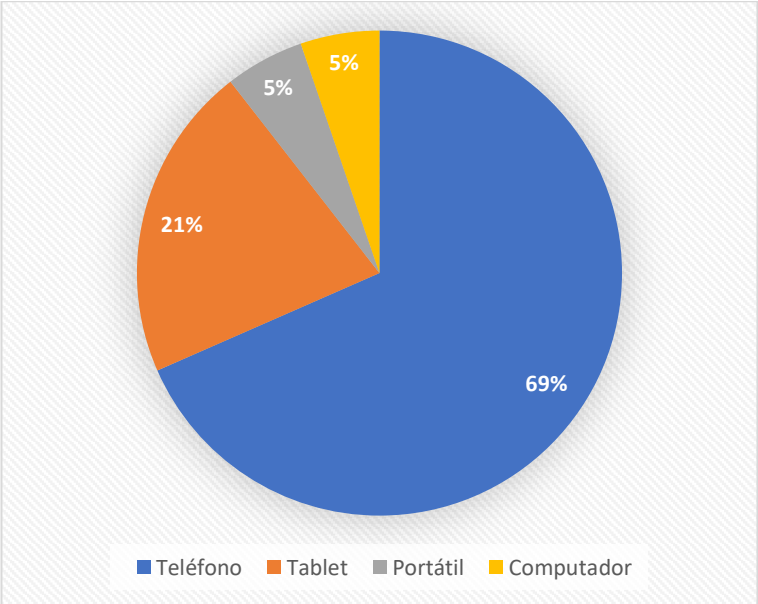


Figura 9. Tabulación de resultados pregunta 9

Análisis e Interpretación

El 69% de los encuestados considera que el teléfono es el dispositivo más fácil de manipular. En contraste, las tablets son preferidas por el 21%. Los dispositivos portátil y computador de escritorio tienen una baja preferencia, cada uno con el 5%. La tendencia indica una clara inclinación hacia la facilidad de uso de los teléfonos por parte de la población.

Pregunta 10: ¿Ha utilizado aplicaciones móviles en dispositivos Android previamente?

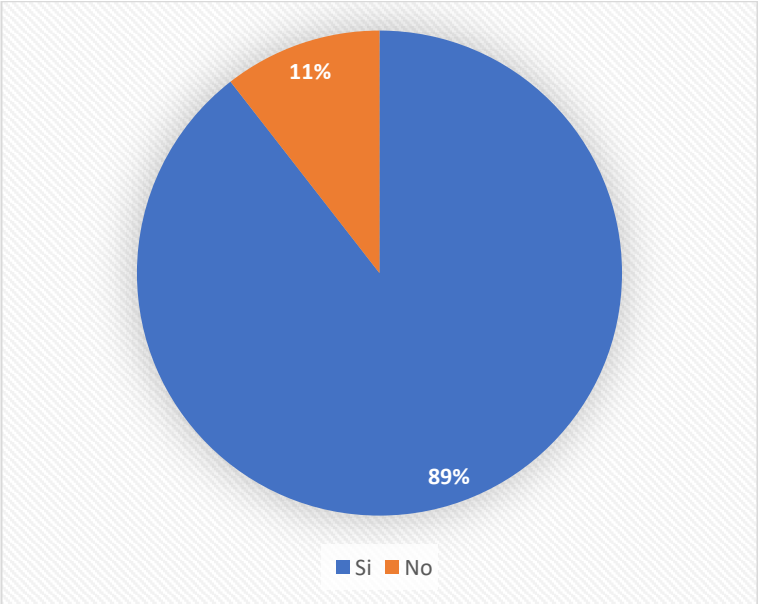


Figura 10. Tabulación de resultados pregunta 10

Análisis e Interpretación

Los resultados de la pregunta 10 demuestran que casi toda la población conoce y ha utilizado alguna vez alguna aplicación móvil en un dispositivo Android, siendo el 89% quienes lo han hecho, mientras que el 11% no lo ha hecho, no necesariamente por el hecho de que nunca hayan tenido acceso a un teléfono inteligente, sino que puede que utilicen únicamente dispositivos con un sistema operativo diferente a Android.

Pregunta 11: ¿Cree que una aplicación móvil facilitaría la gestión de pedidos y mejoraría su desempeño laboral?

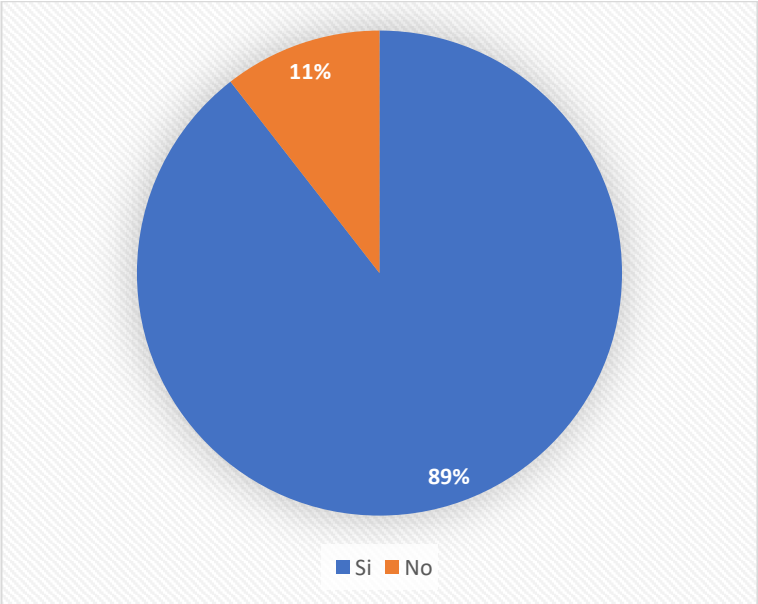


Figura 11. Tabulación de resultados pregunta 11

Análisis e Interpretación

Los resultados de la pregunta 11 muestran que el 89% de los encuestados cree que una aplicación móvil facilitaría la gestión de pedidos y mejoraría su desempeño laboral. Por otro lado, solo un 11% opina que no sería de ayuda. Esto indica una inclinación por el uso de herramientas digitales en el ámbito laboral de esta muestra de personas.

Pregunta 12: ¿Considera que una aplicación móvil para gestionar pedidos podría reducir los errores en la toma y preparación de pedidos?

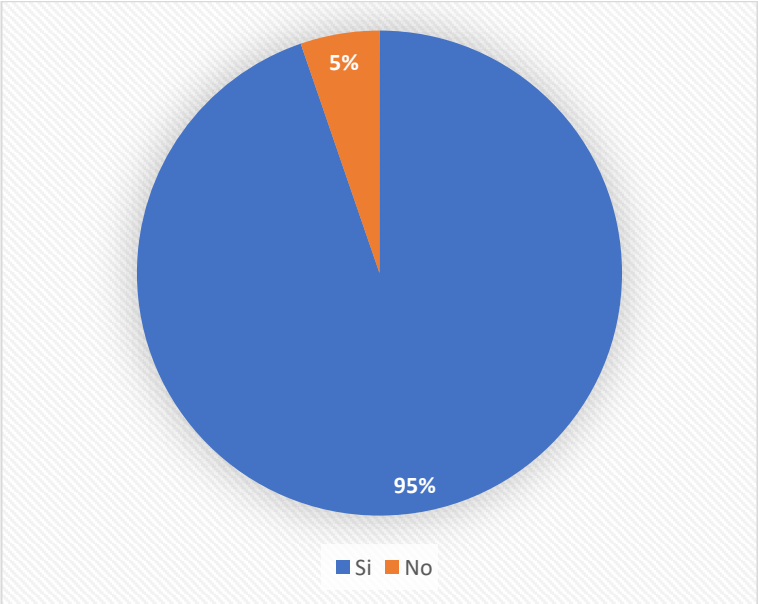


Figura 12. Tabulación de resultados pregunta 12

Análisis e Interpretación

Los resultados de la pregunta 12 demuestran que el 95% de los encuestados considera que una aplicación móvil para gestionar pedidos podría reducir los errores en la toma y preparación de pedidos, mientras que solo el 5.26% opina lo contrario. Con esto se evidencia que se pueden evitar malentendidos que surjan al preparar los mismos, siendo esto una de las causas para la insatisfacción del cliente.

Pregunta 13: ¿Qué tan dispuesto/a estaría a utilizar una aplicación móvil para gestionar pedidos en su trabajo?

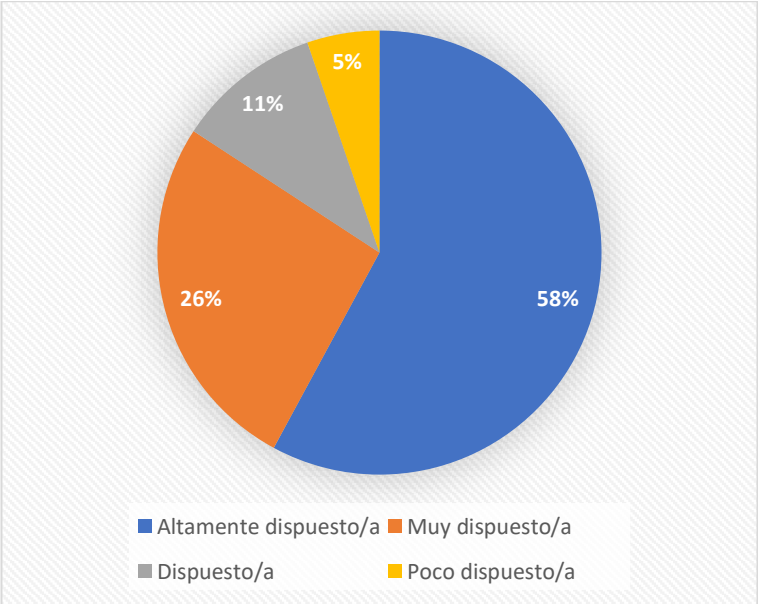


Figura 13. Tabulación de resultados pregunta 13

Análisis e Interpretación

Los resultados de la pregunta 13 demuestran que el 58% de los encuestados está altamente dispuesto a utilizar una aplicación móvil para gestionar pedidos en su trabajo, un 26% está muy dispuesto, un 11% simplemente está dispuesto y solo el 5% está poco dispuesto. Estos resultados indican una predisposición positiva generalizada hacia la adopción de tecnologías móviles en la gestión de pedidos.

Pregunta 14: ¿Le gustaría que se implantara una solución tecnológica para gestionar de manera eficiente los pedidos en Ramones Restaurant?

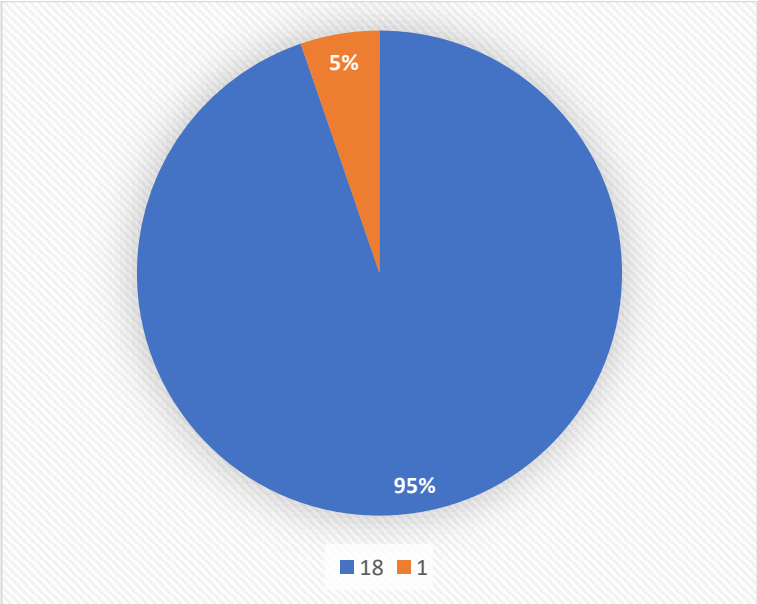


Figura 14. Tabulación de resultados pregunta 14

Análisis e Interpretación

Los resultados de la pregunta 14 indican que el 95% de los encuestados les gustaría que se implantara una solución tecnológica para gestionar de manera eficiente los pedidos en Ramones Restaurant, mientras que solo el 5.26% no lo desea. Estos datos reflejan un claro interés en la adopción de herramientas tecnológicas para mejorar la gestión de pedidos y realizar cambios en la gestión actual.

2.2.4 Procesamiento y análisis de datos

Una vez realizado el respectivo análisis de la técnica aplicada, se puede concluir que:

- La gran mayoría de los empleados, el 89% para ser exacto, ha experimentado dificultades en la coordinación y comunicación en el proceso de pedidos, lo que señala un problema en la gestión actual de pedidos que se lleva a cabo en Ramones Restaurant.

- Los retrasos en los pedidos son comunes, porque un 42% de la población afirma haberlos experimentado a menudo y un 10% siempre. Esto indica que la eficiencia del proceso de pedidos podría mejorarse significativamente.

- Aunque un 37% de los empleados se siente satisfecho con la rapidez en la entrega de pedidos, un 21% está insatisfecho, lo que indica margen para mejoras en la velocidad de servicio.

- El acceso a la información relevante sobre pedidos es un obstáculo para más del 60% de los empleados, lo que afecta la atención al cliente y posiblemente conduce a más errores y retrasos.

- Las áreas más críticas para la mejora son la toma de pedidos, con un 37% de sugerencias para mejoras, y la coordinación interna, con un 32%, seguidas por la entrega y preparación de pedidos.

- Hay un consenso unánime del 100% entre la población en que la gestión de pedidos puede y debe mejorarse para ofrecer un servicio más rápido y efectivo a los clientes.

- Hay una predisposición positiva hacia la adopción de tecnología, especialmente aplicaciones móviles, para mejorar la gestión de pedidos; el 89% cree que facilitaría su trabajo y el 95% está a favor de implementar una solución tecnológica.

CAPÍTULO III. RESULTADOS Y DISCUSIÓN

3.1 Análisis y discusión de resultados

3.1.1 Análisis de procesos

Una vez recogida la información sobre el proceso de gestión de pedidos, se identificaron varios subprocesos en medio, todos primordiales de inicio a fin.

a. Toma de pedidos

- ***Proceso Actual***

El servicio empieza con la toma de pedidos, representado en la Figura 15.

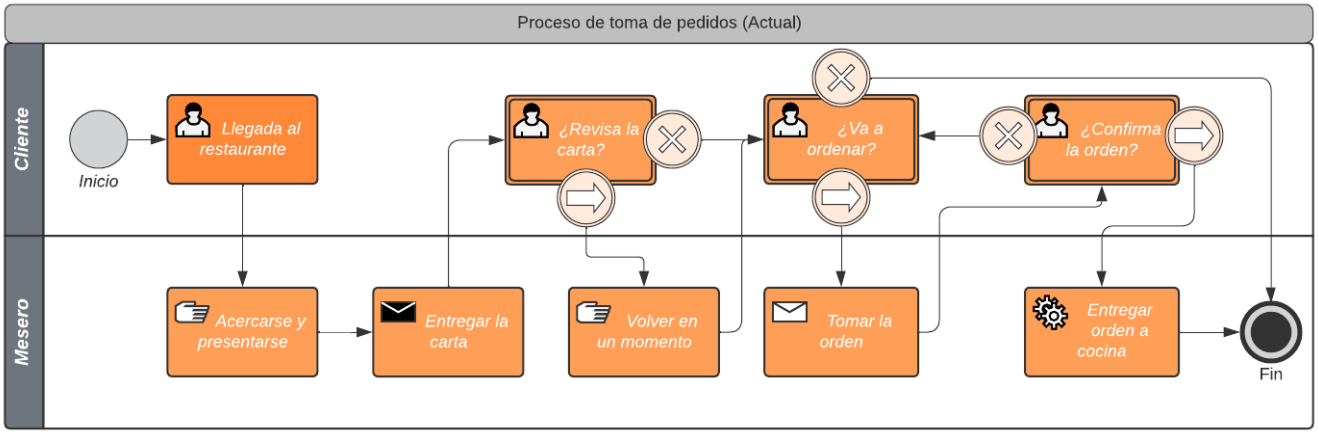


Figura 15. Proceso actual de toma de pedidos

La Figura 15 demuestra el proceso de toma de pedidos que está implantado actualmente en el negocio, de manera sintetizada:

- El cliente llega al restaurante, y se instala.
- El mesero se acerca al cliente y se presenta.
- El mesero hace entrega de la carta al cliente para que la revise.
- El cliente decide si revisar la carta o no, si es así, el mesero vuelve después de un momento, el suficiente para que el cliente tome una decisión, si no, se le

pregunta si va a ordenar, si es así, se procede a tomar su orden, ya que el cliente puede tener una elección de antemano, si no, se termina con el proceso.

- Una vez tomada la orden, se confirma la misma con el cliente, para asegurarse de que todo está correcto, si es así, el mesero se dirige a cocina a entregar la comanda, si no, se vuelve a preguntar si va a ordenar, si es así, se vuelve a tomar la orden a modo de corrección, hasta que el cliente confirme la orden, si no, se termina con el proceso.

- ***Proceso Optimizado***

Una vez analizado el proceso de toma de pedidos, se constató que, de la forma tradicional, se deja un tiempo más prolongado para que el cliente tome una decisión, desde el primer contacto con el mesero, y aunque contradictoriamente, agregar más pasos a este proceso puede ayudar a tomar otra ruta más directa para completar el objetivo.

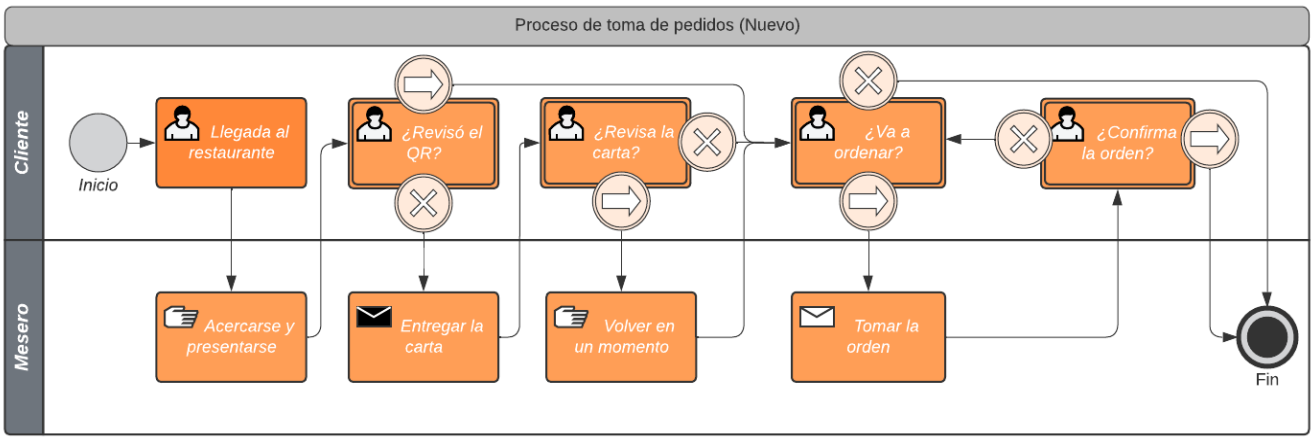


Figura 16. Proceso optimizado de toma de pedidos

En la Figura 16 se evidencia que uno de los cambios más importantes es la supresión de la tarea por parte del mesero de acercarse a cocina a entregar la comanda, ya que, si bien esta se comunica a cocina, el mesero ya no tiene la necesidad de hacerlo personalmente, ya que esta acción se realiza automáticamente a través de la aplicación, una vez se haya confirmado la orden.

Además, al agregar el paso de revisión del código QR para acceder más fácilmente a la carta se logra ofrecer más opciones y flexibilidad, permitiendo un margen más

amplio de tiempo para que el cliente tome una decisión mejor planificada, lo que podría mejorar la experiencia de este.

b. Preparación de pedidos

- ***Proceso Actual***

Respecto a la preparación de los pedidos, el proceso se retoma desde la última tarea realizada anteriormente. En esta etapa, el mesero juega un rol clave al involucrarse directamente para obtener información sobre el estado de preparación del pedido, el tiempo estimado restante, entre otros detalles importantes. Esta información es esencial para que el mesero pueda comunicarse efectivamente con el cliente, manteniéndolo informado y al tanto del progreso de su pedido.

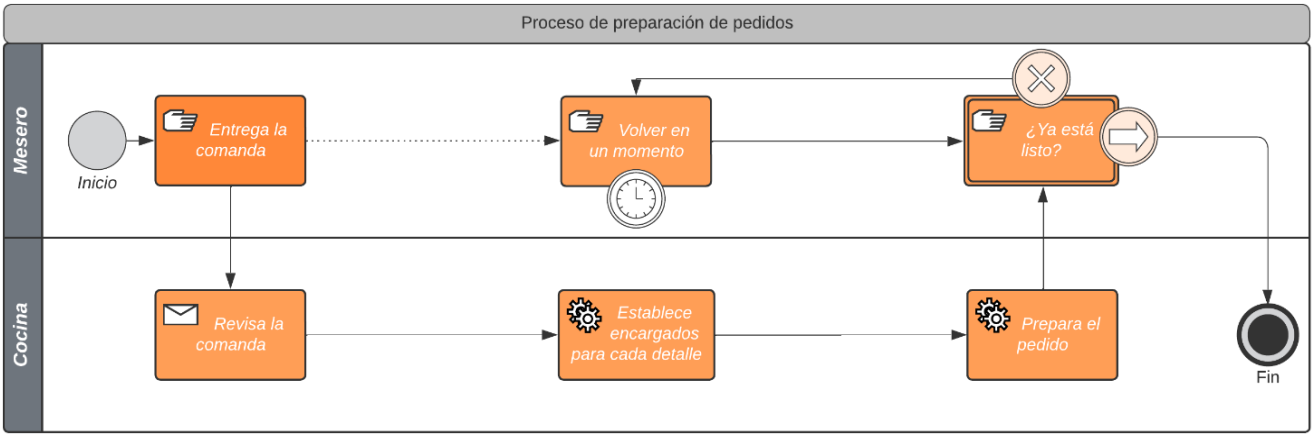


Figura 17. Proceso actual de preparación de pedidos

La Figura 17 demuestra el proceso el proceso de preparación de pedidos que está implantado actualmente en el negocio, de manera sintetizada:

- El mesero entrega la comanda, y se retira por un momento a seguir con sus labores.
- Cocina recibe la comanda.
- Cocina revisa la comanda, analizando cantidades, indicaciones y demás detalles.
- Cocina designa los detalles a diferentes cocineros, en caso de ser una comanda grande.

– Cocina procede a preparar el pedido, desde su cocción (de ser el caso), hasta el emplatado.

– El mesero se acerca a cocina para saber si está listo el pedido, si es así, se termina el proceso de preparación, si no, vuelve en otro momento, hasta que el pedido ya esté servido.

- ***Proceso Optimizado***

Tradicionalmente, la presencia y movilidad del mesero en la cocina son requeridas para tener conocimiento acerca del pedido en curso, ya que no se tiene la certeza de en qué momento exacto el pedido ya está listo, obligándolo a dejar de lado otras actividades que requieran de su atención.

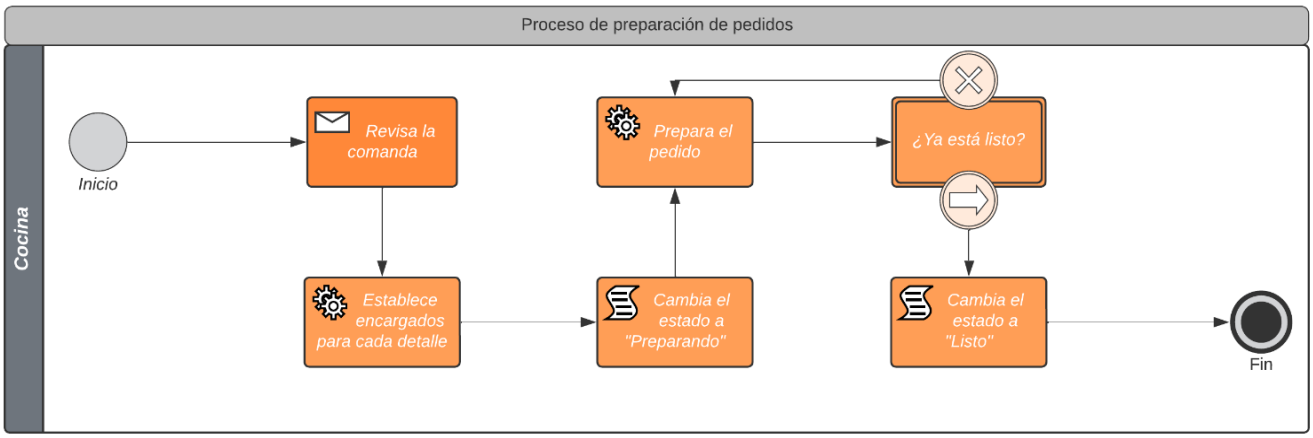


Figura 18. Proceso optimizado de preparación de pedidos

Para el caso de la preparación de los pedidos, acorde a la Figura 18, una de las formas para optimizar el proceso se logra por medio de la aplicación, simplificando la comunicación y traslado a cocina por parte del mesero las veces que crea pertinentes, gracias a la información en tiempo real del estado del pedido, y la notificación instantánea de cuando el pedido está listo, de modo que ya no tiene que acercarse a la cocina innecesariamente.

c. Entrega de pedidos al cliente

No se aplicaron cambios a este proceso, ya que su propia naturaleza lo obliga a ser tan sencillo y rígido, retirar el pedido de cocina y entregarlo al cliente, y de ser el caso, atender una nueva solicitud.

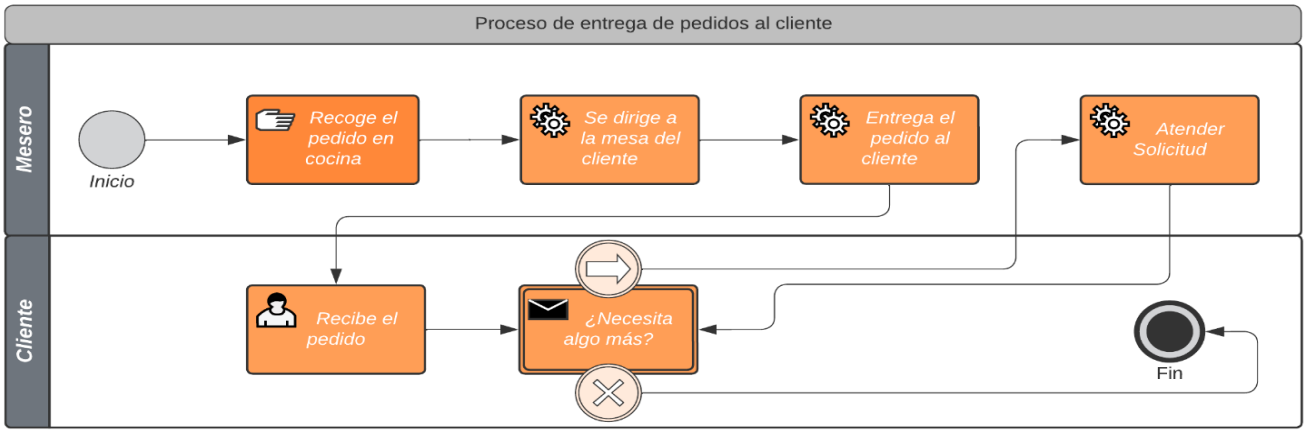


Figura 19. Proceso de entrega de pedidos

En medio de este proceso, se puede deducir que, al atender algún nuevo requerimiento por parte del cliente, se puede tratar de alguna queja o malentendido en su pedido, o también de algún producto extra que el mismo requiera en ese momento, por lo que puede remitirse implícitamente al proceso de toma de pedidos nuevamente.

d. Proceso de cobro al cliente

Este proceso tampoco requirió de cambios en su estructura, ya que su finalidad es la misma, cancelar por todo lo consumido, sin embargo, cabe resaltar que de la forma tradicional, se daba pie a confusiones y malos entendidos debido a que al trabajar con comandas, la búsqueda se debía realizar de forma manual, y la verificación de que el pedido en cuestión se tratase del pedido del cliente, llegaba incluso a requerirse de la presencia del mesero que dio la atención para poder corroborar que fue lo que sirvió. Ahora, con la aplicación la información está correctamente indexada y asociada al número de la orden del cliente, y en base a eso se accede a todo el detalle de la orden, el mesero, la hora, los platillos, cantidades y total a cancelar, sin la necesidad de calcular todo esto de antemano por parte del cajero.

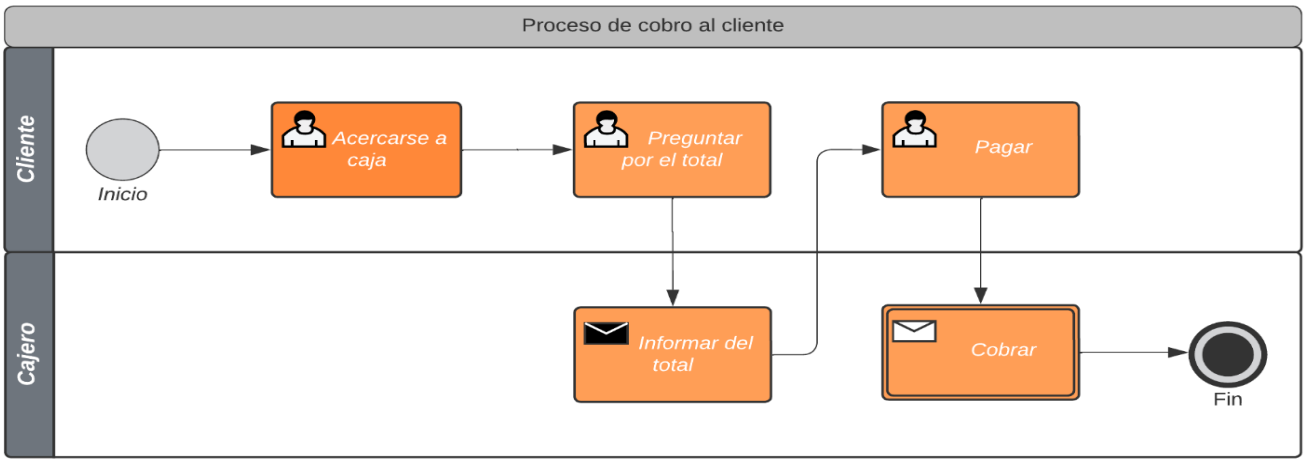


Figura 20. Proceso de cobro

3.1.2 Modelos de predicción

a. Métricas para la optimización del servicio

Acorde a las actividades que se realizan en la gestión de pedidos, y en base a su flexibilidad es que se afirma que se pueden mejorar continuamente, sin embargo, para lograr esto es crucial recopilar y analizar cierto tipo de información y datos concernientes al hecho del negocio, como:

- ***Preferencias y Hábitos de los Clientes***

Recopilar datos sobre los platos más pedidos, los tiempos de espera aceptables, y las preferencias de personalización.

- ***Tiempos de Preparación y Entrega***

Registrar y analizar los tiempos desde que se toma el pedido hasta que se entrega al cliente, identificando cuellos de botella y oportunidades de eficiencia.

- ***Rendimiento del Personal***

Evaluar el rendimiento del equipo en términos de velocidad, precisión y calidad del servicio al cliente.

Todas estas áreas de mejora se pueden tratar partiendo de los datos obtenidos a través de los históricos del negocio, en tablas de seguimiento, libro de registro de comentarios, encuestas de satisfacción del cliente, entre otras, permitiendo una visualización clara del desempeño y facilitando la toma de decisiones estratégicas para mejoras operativas y de servicio. No obstante, aunque efectivos, estos medios requieren de capacitación y conocimiento para poder ser implementados in situ, más aún cuando del análisis se trata, es por eso que actualmente se recurre a alternativas más viables en cuanto a capacidad, análisis y procesamiento de la información se refiere, ya que si bien en base a resultados obtenidos del procesar dicha información de modo manual se puede llegar a una conclusión, se requiere manejar volúmenes de datos muy grandes, además de contar con una estructura de datos bien diseñada para segmentar eficientemente la información, entre otras, y para ello existen algoritmos que tienen la finalidad de asistir a la toma de decisiones, basándose en una arquitectura y un modelo construido para dar solución a un problema específico, los denominados modelos de predicción, también conocido como machine-learning, una rama de la inteligencia artificial.

Tabla 3. Tipos de aprendizaje automático

Criterios	**Aprendizaje Supervisado**	**Aprendizaje no supervisado**	**Aprendizaje por refuerzo**
Tipo de datos	Etiquetados	No etiquetados	Interacción con el entorno
Objetivo	Predicción/Clasificación	Descubrimiento de patrones	Aprendizaje de decisiones
Ejemplo de modelos	Regresión, Redes Neuronales	Clustering, PCA	Q-Learning, Deep Q-Networks
Aplicaciones	Pronóstico, diagnóstico	Segmentación de mercado	Juegos, Navegación autónoma

En base a la Tabla 3, los algoritmos que mejor se acoplan a los objetivos del proyecto son de tipo aprendizaje supervisado y no supervisado, por lo que se realizó un análisis de este tipo de algoritmos.

Analizando las características de los modelos, por medio de la siguiente tabla se puede clasificar de mejor manera la factibilidad de los modelos en base a criterios fundamentales para el proyecto:

Tabla 4. Comparativa entre modelos de predicción propuestos

Criterio	Regresión Lineal	Árboles de Decisión	Máquinas de Soporte Vectorial	Redes Neuronales	Series Temporales	Clustering
Facilidad de Implementación	Alta	Alta	Media	Baja	Media	Media
Precisión	Media	Alta	Alta	Muy Alta	Alta	Media
Velocidad de Ejecución	Alta	Media	Baja	Baja	Media	Media
Requerimientos de Datos	Bajos	Medianos	Altos	Muy Altos	Medianos	Medianos
Interpretabilidad	Alta	Alta	Baja	Baja	Media	Media
Tipo de Aprendizaje	Supervisado	Supervisado	Supervisado	Mixto	Supervisado	No Supervisado

Una vez analizada la Tabla 4, se optó por realizar un modelo basado en clusterización y una red neuronal, debido a que los datos con los que se diseñará la arquitectura no están etiquetados, por lo que el clustering se encargará de establecer la variable a predecir, y la red neuronal porque los datos no cuentan con una relación lineal, esto resulta idóneo a implantarse en el proyecto debido a la flexibilidad de ambos modelos, además de los resultados y predicciones que puede ofrecer, también por su costo computacional no tan elevado, lo cual en el entorno del negocio resulta crucial por la nula cantidad de datos históricos que deben ser generados a partir de la aplicación móvil para poder entrenar al modelo.

3.1.3 Desarrollo de aplicaciones móviles

Las aplicaciones móviles se han convertido en una parte integral de la vida diaria, ofreciendo una accesibilidad sin precedentes y una comodidad inigualable. Facilitan la conexión instantánea con una amplia gama de productos y servicios, independientemente del lugar o el momento, transformando así la forma en que se interactúa con el mundo. Su capacidad para adaptarse a diversas necesidades y preferencias personales las hace no solo útiles, sino también esenciales en el dinámico entorno actual.

Para desarrollar una aplicación móvil, se requiere una combinación de habilidades técnicas, además de la selección de una plataforma y herramientas necesarias para su construcción, basándose en la funcionalidad y el propósito que se ha definido para la aplicación. Esto se logra a través de una planificación cuidadosa y una investigación del mercado, todo dentro del marco de cumplimiento de los objetivos gracias a la implementación de una metodología de desarrollo.

a. Plataformas de desarrollo

Las plataformas de desarrollo móvil son entornos en los que los desarrolladores crean, prueban y despliegan aplicaciones destinadas a dispositivos móviles. Cada plataforma tiene sus propias herramientas, lenguajes de programación y entornos de desarrollo integrados (IDEs), y está dirigida a sistemas operativos específicos como Android, iOS, o incluso múltiples sistemas a través de enfoques de desarrollo cruzado.

Tabla 5. Plataformas de desarrollo móvil

Plataforma	Lenguaje Principal	Mercado	IDEs o Frameworks
Android	Java, Kotlin	Mayor cuota de mercado global	Android Studio
iOS	Swift, Objective-C	Mercado Premium, alto poder adquisitivo	Xcode
Cross-Platform	Javascript, Dart, C++, TypeScript	Amplia compatibilidad	React Native, Flutter, Xamarin, Ionic

Acorde a la Tabla 5, se optó por Android como plataforma principal para el desarrollo de aplicaciones móviles, con un enfoque en soluciones cross-platform, es una decisión estratégica a largo plazo. Android, con su amplia cuota de mercado y naturaleza de

código abierto, facilita la personalización y adaptabilidad a diversos dispositivos y usuarios. Esta elección, al ser compatible con múltiples plataformas, permite expandirse a otros sistemas operativos en el futuro, aumentando así el alcance potencial de la aplicación.

b. Frameworks frontend y backend

Al seleccionar una solución cross-platform, se da la pauta a hacer uso de Frameworks para desarrollo frontend y backend, elementos clave en la creación de aplicaciones modernas. En el ámbito del frontend, frameworks como Angular, React, Vue.js, Ionic, y Flutter son cruciales. En el desarrollo backend, frameworks como Node.js, Django, y .NET son fundamentales.

Analizando los Frameworks para frontend, se resume en:

Tabla 6. Frameworks para desarrollo frontend

Framework	**Lenguaje**	**Flexibilidad**	**Documentación**	**Facilidad de Implementación**
Angular	TypeScript	Alta	Excelente	Media
React	JavaScript	Muy Alta	Excelente	Alta
Vue.js	JavaScript	Alta	Muy Buena	Alta
Ionic	HTML, CSS, JavaScript	Alta	Excelente	Alta
Flutter	Dart	Media	Muy Buena	Media

Conforme a la información resumida en la Tabla 6, se escogió a Ionic como framework para el desarrollo frontend, combinado con Angular, por las interfaces atractivas que se puede lograr con su material. En definitiva, Ionic es ideal para aplicaciones móviles híbridas o web progresivas, lo cual brinda más opciones de alcance multiplataforma a futuro.

En cuanto a los Frameworks para backend, se obtiene que:

Tabla 7. Frameworks para desarrollo backend

Framework	**Lenguaje**	**Flexibilidad**	**Documentación**	**Facilidad de Implementación**	**Mecanismos de Seguridad**	**Base de Datos recomendada**
Node.js	JavaScript	Muy Alta	Muy Buena	Alta	Autenticación JWT, OAuth, CORS, XSS y CSRF Protección	MongoDB (NoSQL), MySQL
Django	Python	Alta	Excelente	Media	Autenticación de usuarios, Protección CSRF, Seguridad en contraseñas, Control de Acceso	PostgreSQL (por su robustez y características avanzadas)
.NET	C#	Alta	Excelente	Alta	Autenticación y autorización, Protección de datos, Seguridad en ASP.NET	SQL Server (por su integración y optimización con .NET)

De la Tabla 7 se seleccionaron dos Frameworks, uno para toda la interacción entre la aplicación y la comunicación con la base de datos, y otro para la ejecución del modelo de predicción, .NET y Django respectivamente, con su enfoque en C#, permite un desarrollo robusto y eficiente, y con Python se simplifica la generación e implementación de algoritmos complejos.

c. Bases de Datos

Una base de datos es un sistema organizado para almacenar, gestionar y recuperar información de manera eficiente. Funciona como un repositorio centralizado donde se pueden guardar y acceder a grandes cantidades de datos de forma estructurada. Las bases de datos son fundamentales en casi todos los sistemas informáticos, desde sitios web y aplicaciones móviles hasta sistemas empresariales complejos.

En base a la Tabla 7, se seleccionó SQL Server como base de datos al ser la recomendada, especialmente cuando se trabaja con .NET en el backend. SQL Server, desarrollado por Microsoft, se integra de manera óptima con el ecosistema de Microsoft, lo que garantiza una mayor eficiencia y cohesión en el desarrollo y gestión de aplicaciones. Esta integración proporciona ventajas como una mejor gestión de datos, seguridad reforzada y una compatibilidad sin fisuras, haciendo que SQL Server sea una opción sólida y confiable para proyectos que utilizan tecnologías de Microsoft.

d. Metodologías de desarrollo móvil

Las metodologías de desarrollo móvil son conjuntos de prácticas y procedimientos diseñados específicamente para la creación y mantenimiento de aplicaciones móviles. Estos enfoques proporcionan marcos de trabajo esenciales para guiar a los desarrolladores a través de los diversos aspectos del desarrollo móvil, desde la conceptualización hasta la implementación y el lanzamiento de la aplicación. Al enfocarse plenamente en una planificación eficiente, estas metodologías buscan garantizar la calidad del producto final, la satisfacción del usuario y la agilidad en la respuesta a los cambiantes requisitos del mercado. A continuación, se explorarán algunas de las metodologías más aptas para la ejecución del proyecto, en el ámbito del desarrollo de aplicaciones móviles.

Las metodologías más acordes para implementar de acuerdo con la naturaleza y requerimientos del proyecto son:

Tabla 8. Metodologías de desarrollo móvil

Metodología	**Enfoque**	**Tamaño del Equipo**	**Flexibilidad**	**Planificación**	**Idoneidad para Proyectos Individuales**	**Enfoque en la Calidad**	**Comunicación con el Cliente**	**Documentación General del Proyecto**
Mobile-D	Móvil	Pequeños	Alta	Iterativa e incremental	Moderada	Alta	Regular	Detallada
Scrumban	Híbrido	Flexible	Muy Alta	Flexible y adaptativa	Alta	Moderada	Flexible/Adaptable	Moderada
Scrum	Ágil	Pequeños a Medianos	Media	Sprints fijos	Moderada	Alta	Regular/Intensa	Detallada
Extreme Programming (XP)	Ágil	Pequeños	Alta	Iterativa con lanzamientos frecuentes	Moderada	Muy Alta	Muy Intensa	Detallada

En base a la Tabla 8, Scrumban resulta una metodología aplicable debido a que combina la flexibilidad de Kanban con la estructura de Scrum, lo que la hace muy adaptable y adecuada para proyectos individuales, con una planificación más flexible y adaptativa, además de que la comunicación con el cliente es adaptable dependiendo de la disponibilidad de este, lo que la hace adecuada para proyectos individuales y dinámicos.

3.2 Desarrollo de la propuesta

Acorde a la metodología seleccionada para el desarrollo de la solución, el proceso completo se divide en 7 fases comúnmente, sin embargo, al aportar con la flexibilidad a cambios permite simplificar la longitud de estas, e incluso suprimir ciertas fases, teniendo en cuenta que el equipo solo lo conforma el investigador.

3.2.1 Fase 1: Establecer los objetivos

Los objetivos del desarrollo están alineados con los objetivos del proyecto, por lo que en resumen trabajando en la propuesta, se atiende a las necesidades del negocio, que detalladamente vendrían a ser:

- **Optimización del servicio:** Con la solución, la forma en la que se da la interacción entre los diferentes actores del sistema es más clara y simplificada, lo que resulta en una carga de trabajo equilibrada.

- **Gestión de la información:** Se refiere a la posibilidad de acceder de forma ordenada y clara a los diferentes servicios brindados por el personal, de modo que se evitan malentendidos que desembocan en quejas e insatisfacción para los usuarios.

- **Toma de decisiones:** La información generada por la solución sirve como punto de partida para establecer análisis y metas, basándose en los históricos se puede llegar a una síntesis de los cambios que se requieran aplicar al negocio, haciendo uso de los modelos de predicción, se destinan los recursos eficientemente, todo en pro de tomar decisiones informadas.

- **Satisfacción del cliente:** Las mejoras que se realizan en las distintas áreas forman parte de un todo, cuya finalidad es la de cumplir los objetivos, y si la organización implementa cambios ya sea en sus procesos, gestión o enfoque general, todos los actores del proyecto se verán beneficiados.

3.2.2 Fase 2: Backlog del proyecto

El backlog representa un listado de todas las tareas por hacer para culminar con el proyecto, para este caso, se definió de forma general a los módulos del sistema, también a los actores del proyecto, quienes harán uso de este respectivamente.

a. Actores del Proyecto

Los actores en un proyecto son los distintos tipos de usuarios o entidades que interactúan con el sistema. Cada actor representa un rol con un conjunto específico de necesidades, comportamientos y formas de interactuar con la aplicación. Los actores principales para este proyecto son:

Tabla 9. Actores principales del proyecto

Actor	Descripción	Funciones Principales
Mesero	Gestiona atenciones y registra pedidos de clientes.	Registrar nuevos pedidos, dar seguimiento a los pedidos realizados.
Personal de Cocina	Visualiza y gestiona el estado de los pedidos de comida.	Gestionar el estado de preparación de los pedidos.
Cajero	Procesa cobros y gestiona pedidos desde el punto de vista financiero.	Cobrar pedidos, visualizar pedidos y su historial.
Administrador	Supervisa y administra las operaciones generales del restaurante.	Administración general, acceso a reportes y configuraciones.

b. Módulos

Viene a representar los requisitos funcionales del sistema, es decir aquellos que dictan el comportamiento y funcionamiento al momento de interactuar con este.

Tabla 10. Módulos del sistema

Módulo	Funcionalidad	Usuario
Login	Inicio de sesión seguro para los empleados con sus credenciales.	Todos los actores
Mesero	Pantalla para ver y registrar pedidos de clientes por parte del mesero.	Mesero
Cocina	Interfaz para visualizar y gestionar el estado de los pedidos en la cocina.	Personal de Cocina

Módulo	Funcionalidad	Usuario
Caja	Funcionalidad para procesar cobros y visualizar pedidos y su historial en la caja.	Cajero
Administración	Interfaz para visualizar reportes de las ventas realizadas, agregar nueva información a la base de datos.	Administrador
Perfil	Sección de perfil para que los empleados vean su información personal.	Todos los actores

c. Requisitos y Tareas del Proyecto

Hace alusión a aquellos requerimientos que forman parte de la solución implícitamente. Los requisitos no funcionales se refieren a los estándares, cualidades y atributos que aseguran el buen funcionamiento y eficiencia de la aplicación, más allá de sus características específicas.

- ***Mejoras***

Mejora de la interfaz de usuario y optimización de la sincronización de datos entre módulos.

- ***Correcciones de errores***

Corrección de errores en la funcionalidad de registro de pedidos.

- ***Tareas técnicas***

Establecimiento de una base de datos segura y eficiente; integración de un sistema de notificaciones, diseño de una arquitectura de red neuronal para realizar predicciones, consumo de servicios con protocolos seguros.

- ***Pruebas***

Realización de pruebas unitarias, de integración y de usabilidad con empleados.

- ***Documentación***

Documentación del desarrollo y creación de manuales de usuario y guías de capacitación.

3.2.3 Fase 3: Planificación de actividades

Cada módulo del sistema se desglosa en distintas actividades necesarias para atender sus requerimientos planteados, e inclusive aceptar unos cuantos nuevos sobre la marcha, de ser el caso.

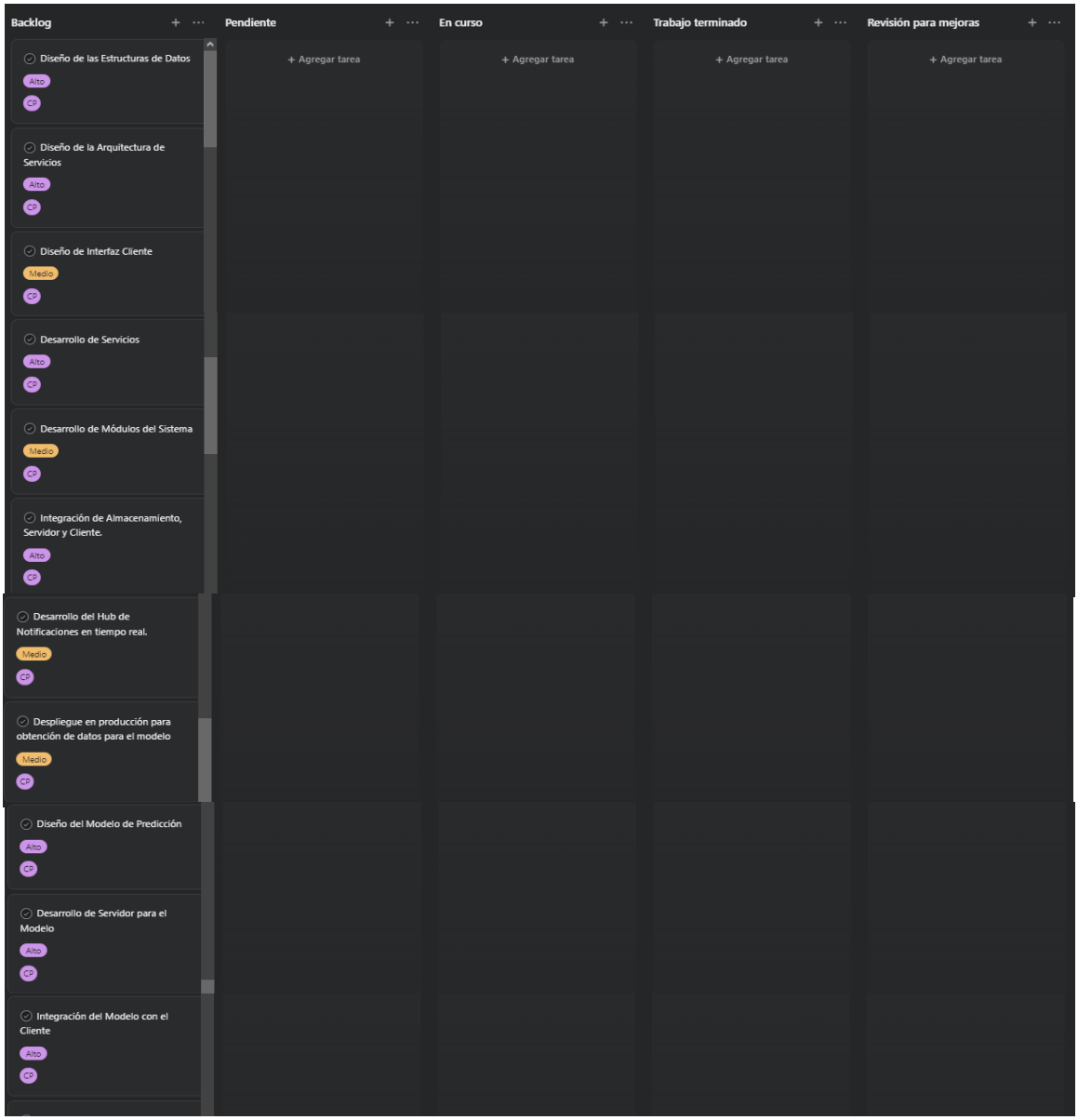


Figura 21. Tablero Kanban de actividades

Para la representación de estas actividades a través del tiempo de ejecución del proyecto, se hizo uso de la herramienta web Asana, dónde se plasman de manera general las actividades necesarias para la culminación del proyecto, representadas en un tablero Kanban según la Figura 21, que consecuentemente serán desglosadas y segmentadas por sprints de desarrollo.

Cada sprint está planificado para desarrollarse en un lapso de 2 semanas máximo, dependiendo de la cantidad de actividades que se cubran en el sprint en cuestión, con la consideración de que, en etapas de desarrollo, se cubre todos los módulos del sistema en sus aspectos correspondientes, backend y frontend, para que el progreso sea uniforme trabajando paralelamente, de modo que, al llegar al final de la solución, la implementación sea manejable.

3.2.4 Fase 4: Desarrollo iterativo

a. Sprint 1

En el primer sprint se cubrieron aspectos fundamentales del inicio del proyecto, que principalmente son tareas de diseño y modelado.

- ***Extracción de características***

Un apartado importante al momento de estructurar la información es analizar e identificar los atributos de cada clase o entidad que forma parte del sistema, e interactúa con él, de modo que al momento de diseñar la base de datos se tengan las características esenciales que se requiera de cada clase.

Columna	Tipo de Dato	Longitud Máxima	precision	scale	Es Nullable	Es Clave Primaria
ID_DET	int	4	10	0	0	1
ID_FACT_PER	int	4	10	0	0	0
ID_PRO	int	4	10	0	0	0
CANT	int	4	10	0	0	0
PU	decimal	9	18	2	0	0
EST_DET	nchar	2	0	0	1	0
LLEV_DET	nchar	2	0	0	1	0
CANT_LLEV_DET	int	4	10	0	1	0
INDIC_DET	nvarchar	2000	0	0	1	0

Figura 22. Diccionario de datos tabla “Detalle_Factura”

- ***Interacción entre las clases***

Se refiere a como interactúa una clase con otra, que tipo de transacción se lleva a cabo entre ellas y que información requieren generar o transmitir para lograr esta interacción, es necesaria para entender la relación que tiene para posteriormente plasmarla en la base de datos. Esta interacción está resumida en el punto 3.1.1.

- ***Diseño de la base de datos***

Basándose en las clases, sus características e interacción entre sí, se diseñó la base de datos en SQL Server que funciona como principal medio de almacenamiento para todas las operaciones y transacciones que se lleven a cabo de acuerdo con el hecho del negocio, en este caso, las ventas que realizan, obteniendo el diagrama entidad relación de la Figura 23.

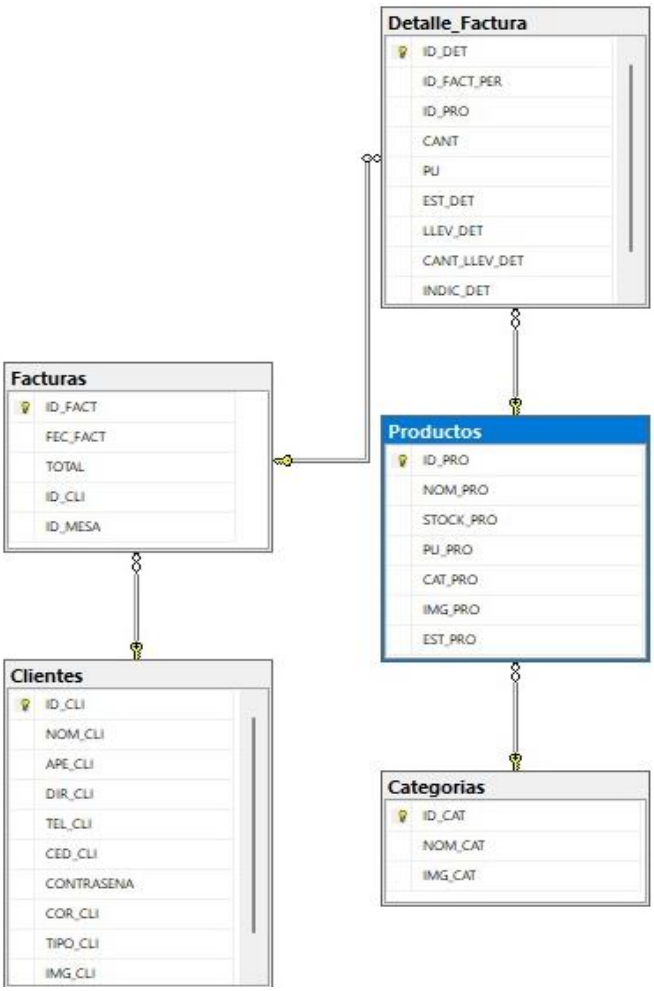


Figura 23. Diagrama Relacional

- ***Creación del proyecto Web API***

.NET ofrece acceso a una amplia gama de herramientas y características para el manejo de multiproyectos, a través de su manejador de paquetes, se logra cubrir distintas problemáticas que surjan en el proceso de desarrollo, con soluciones preestablecidas por la comunidad o por la misma Microsoft, sin necesidad de reinventar la rueda.

Un proyecto Web API es una herramienta que pertenece a .NET MVC y facilita la creación de APIs compatibles con REST. Estas APIs permiten a sistemas externos utilizar la lógica de negocios de una aplicación, mejorando así su reutilización.

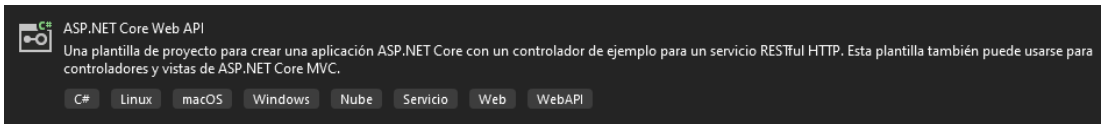


Figura 24. Proyecto .NET Web API

La estructura del proyecto es simple, asemejándose a una arquitectura MVC, que cuenta con distintos apartados como se muestra en la Figura 25.

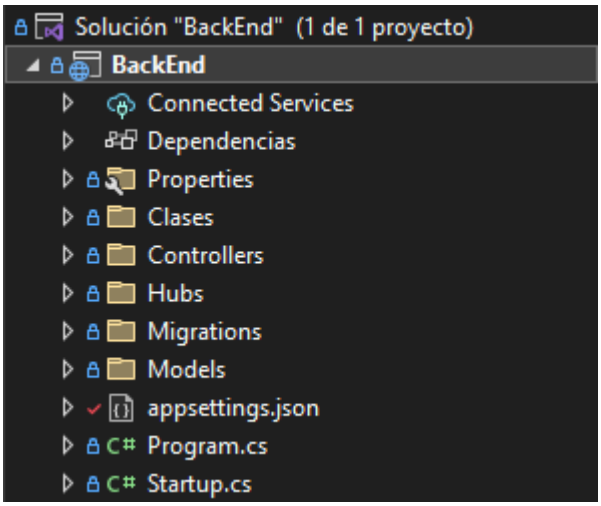


Figura 25. Estructura del backend

- ***Dependencias requeridas***

Las dependencias del proyecto son los archivos de soluciones externas o provistas por el mismo framework que hacen que la aplicación funcione correctamente, como una biblioteca o un complemento. En este contexto, las dependencias utilizadas para el proyecto se detallan en la Figura 26.

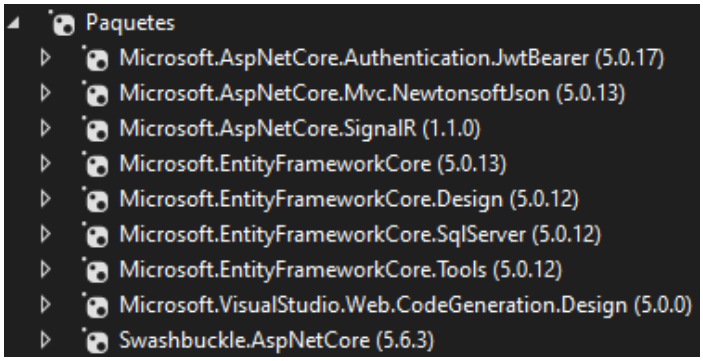


Figura 26. Dependencias del proyecto

- ***Archivo de configuración***

En el archivo Startup.cs se realiza toda la configuración necesaria para la ejecución y correcto funcionamiento de las solicitudes entrantes y salientes, manejo de CORS, entre otros.

En el archivo appsettings.json, se establecen ciertos parámetros y configuraciones extras, como configurar claves, variables, middlewares, entre otros.

Figura 27. Archivo appsettings.json

```
0 referencias | Cristian López, Hace 142 días | 1 autor, 4 cambios
public void ConfigureServices(IServiceCollection services)
{

    services.AddControllers();
    services.AddSwaggerGen(c =>
    {
        c.SwaggerDoc("v1", new OpenApiInfo { Title = "BackEnd", Version = "v1" });
    });
    services.AddDbContext<RamonesDBContext>(options =>
    options.UseSqlServer(Configuration.GetConnectionString("DevConnection")));
    services.AddCors(options => options.AddPolicy("AllowwebApp",
    builder =>
    {
        builder.SetIsOriginAllowed((host) => true)
        .AllowAnyHeader()
        .AllowAnyMethod()
        .AllowCredentials();
    }));

    services.AddControllers().AddNewtonsoftJson(option => option.SerializerSettings.ReferenceLoopHandling = Newtonsoft.Json.ReferenceLoopHandling.Ignore);

    //añadir las configuraciones del json
    var seccionappSetings = Configuration.GetSection("AppSettings");
    services.Configure<AppSettings>(seccionappSetings);

    //jwt
    var appSettings = seccionappSetings.Get<AppSettings>();
    var llave = Encoding.ASCII.GetBytes(appSettings.Llave);

    services.AddAuthentication(d =>
    {
        d.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;
        d.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;

    }).AddJwtBearer(d=>
    {
        d.RequireHttpsMetadata = false;
        d.SaveToken = true;
        d.TokenValidationParameters = new TokenValidationParameters
        {
            ValidateIssuerSigningKey = true,
            IssuerSigningKey = new SymmetricSecurityKey(llave),
            ValidateIssuer = false,
            ValidateAudience = false
        };


    }) ;
    services.AddSignalR();

}
```

Figura 28. Configuración de servicios en Startup.cs

- ***Mockups***

Comprende al maquetado y diseño inicial de las interfaces de los diferentes módulos del sistema, esto ayuda a dar un mejor preámbulo de cómo debe verse el producto final, desde la perspectiva del cliente, además, de esta forma se crea una mejor visión de cómo se dará el funcionamiento en los módulos, como se tratará y mostrará la información.

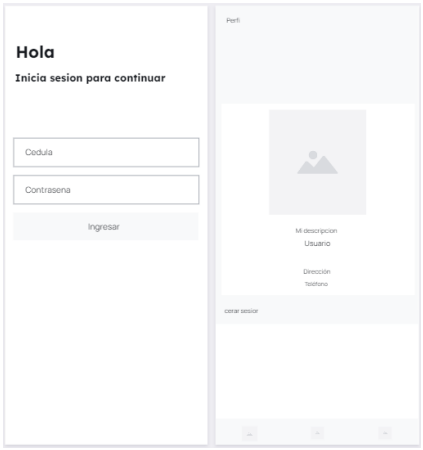


Figura 29. Mockups login y perfil de usuario.

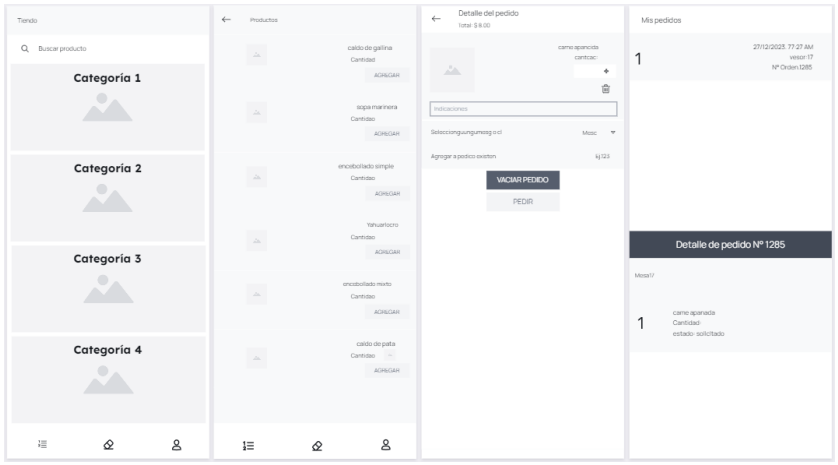


Figura 30. Mockups módulo mesero

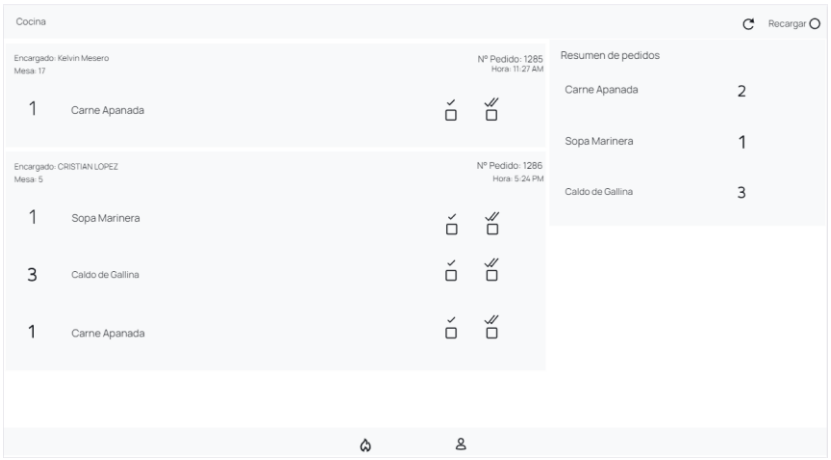


Figura 31. Mockup módulo cocina

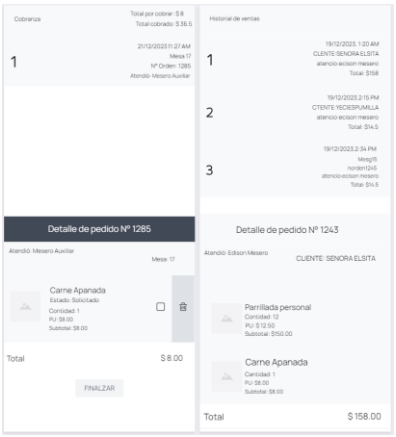


Figura 32. Mockups módulo cajero

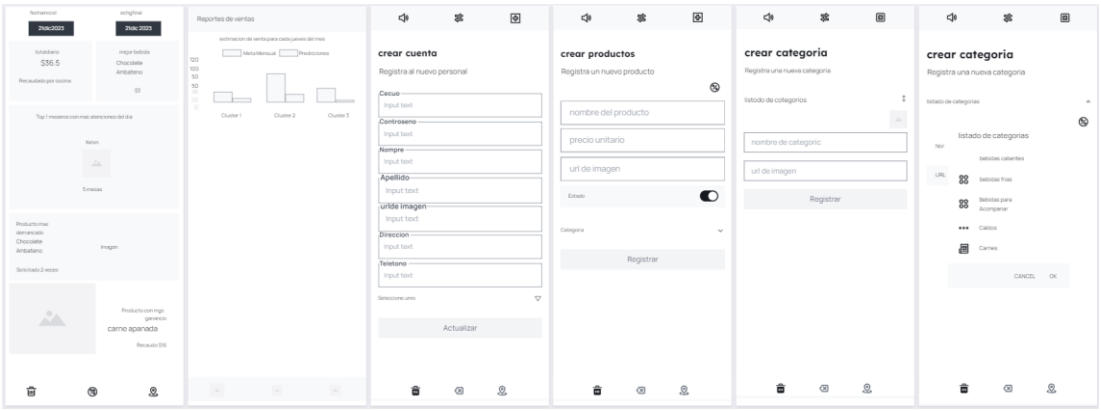


Figura 33. Mockups módulo administrador

- ***Creación del proyecto Ionic-Angular (Frontend)***

Con los mockups se puede proceder a la creación del proyecto para el frontend, que, comprendiendo mejor la visión del cliente, se puede definir mejor la arquitectura de los distintos componentes que forman parte de cada módulo.

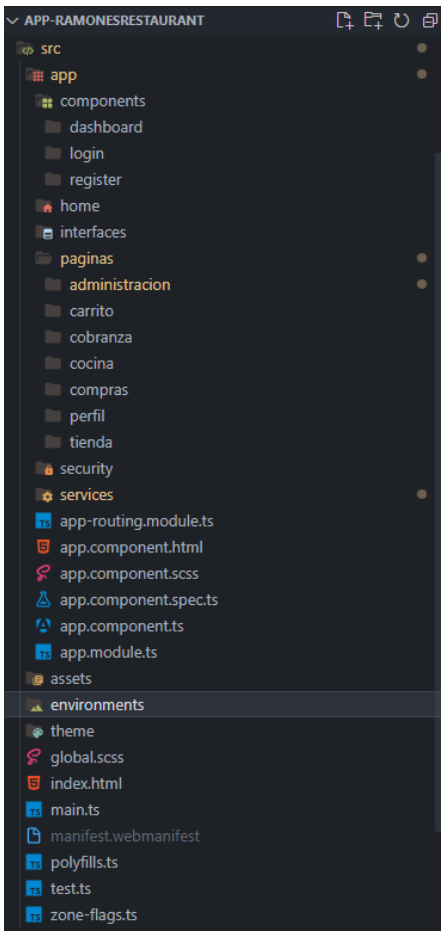


Figura 34. Proyecto Frontend

b. Sprint 2

En este sprint se empezó con el desarrollo inicial tanto de frontend como backend, con la generación de las clases partiendo del catálogo de la base de datos, los controladores necesarios para el login, perfil y consultas a la base de datos necesarias para el usuario, así como la configuración de un protocolo de hashing para la contraseña del usuario, y la generación de un token para la navegación a través del sistema.

- ***Scaffolding***

Se trata del proceso por el cual, a través de la consola de administrador de paquetes, se extrae el contexto de la base de datos para generar el código de las clases, que vienen a ser las tablas, para poder hacer operaciones en ella con sentencias LINQ sencillas, evitando el uso directo de lenguaje SQL.

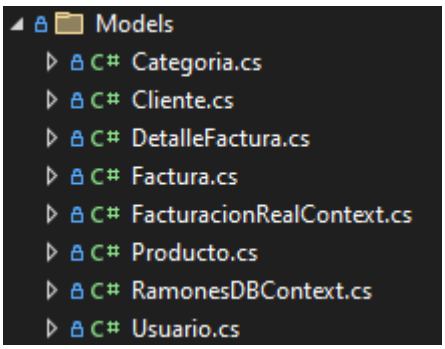


Figura 35. Modelos de la base de datos

- ***Login – Backend***

Con las clases generadas, se procedió a crear los controladores y clases extras necesarias para cubrir el proceso de login.

```
[HttpPost]
[Route("authcliente")]
0 referencias | Cristian López, Hace 161 días | 1 autor, 1 cambio
public async Task<IActionResult> GetClienteAutenticado(Clases.Login credenciales)
{
    Autenticar autenticar = new Autenticar();
    var cliente = await _context.Clientes.Where(user => user.CedCli == credenciales.Usuario).FirstOrDefaultAsync<Cliente>();
    if (cliente == null || cliente.Contrasena == null)
    {
        return BadRequest(new { message = "Usuario no encontrado" });
    }
    else
    {
        var contrasenha = Encriptar.GetSHA256(credenciales.Contrasena);
        if (cliente.Contrasena.Equals(contrasenha))
        {
            autenticar.usuario = cliente;
            autenticar.token = GetToken(cliente.IdCli);
            return Ok(autenticar);
        }
        return BadRequest(new { message = " Contraseña incorrecta" });
    }
}
#endregion
2 referencias | Cristian López, Hace 161 días | 1 autor, 1 cambio
private string GetToken(int id)
{
    var tokenHandler = new JwtSecurityTokenHandler();
    var llave = Encoding.ASCII.GetBytes(_appSettings.Llave);
    var tokenDescriptor = new SecurityTokenDescriptor
    {
        Subject = new ClaimsIdentity(
            new Claim[]
            {
                new Claim(ClaimTypes.NameIdentifier, id.ToString())
            }
            ),
        Expires = DateTime.UtcNow.AddDays(60),
        SigningCredentials = new SigningCredentials(new SymmetricSecurityKey(llave), SecurityAlgorithms.HmacSha256Signature)
    };
    var token = tokenHandler.CreateToken(tokenDescriptor);
    return tokenHandler.WriteToken(token);

}
```

Figura 36. API de autenticación

- ***Login – Frontend***

Con los mockups, se puede empezar a estructurar las vistas mucho más rápido y con una mejor comprensión del funcionamiento esperado. Para el caso del login, se comprende que datos se van a tratar, la respuesta esperada, y a futuro el cómo se llevará a cabo esta operación con el backend.

```
ingresar() {
  let user = this.form.value.usuario.toString();
  if(user.length === 9){
    user = user.padStart(10, '0');
  }
  const password = this.form.value.password;
  const credenciales: Login = {
    usuario: user,
    contrasena: password
  }
  var correcto = null;
  this._loginService.autenticarUsuario(credenciales)
    .subscribe(data => {
      correcto = data;
      if (correcto != null) {
        this.router.navigate(['/dashboard']);
      }
    }, async error => {
      const toast = await this.toastController.create({
        message: error.message,
        duration: 2000
      });
      toast.present();
      console.log(error)
    })

}
```

Figura 37. Función de login frontend.

c. Sprint 3

A partir del login, se procedió con los apartados principales del sistema, una vez se tiene cubierto aspectos de seguridad en la navegación, se construyeron los controladores para las operaciones que ameriten realizar consultas con la base de datos, y a su vez estructurar las vistas en el frontend. Para ello se crearon servicios que consuman las APIs definidas en los controladores, además para establecer una comunicación bidireccional se desarrolló un hub para notificaciones usando SignalR, el cual permite comunicar cambios o actualizaciones que ocurren entre distintas vistas en dependencia de las acciones realizadas por ellas.

Por motivos de optimización, se mostrará el código solo de aquellas operaciones importantes que formen parte del hecho del negocio, es decir las ventas y lo estrictamente relacionado a estas.

- ***Hub de notificaciones***

El hub de notificaciones representa una clase que será llamada con distintos mensajes en las APIs definidas en los controladores, y declarada como endpoint en el archivo de configuración Startup.cs.

```
3 referencias | Cristian López, Hace 148 días | 1 autor, 1 cambio
public class MiHub : Hub
{
    0 referencias | 0 cambios | 0 autores, 0 cambios
    public async Task Notificar(string mensaje)
    {
        await Clients.All.SendAsync("Notifación", mensaje);
    }
}
```

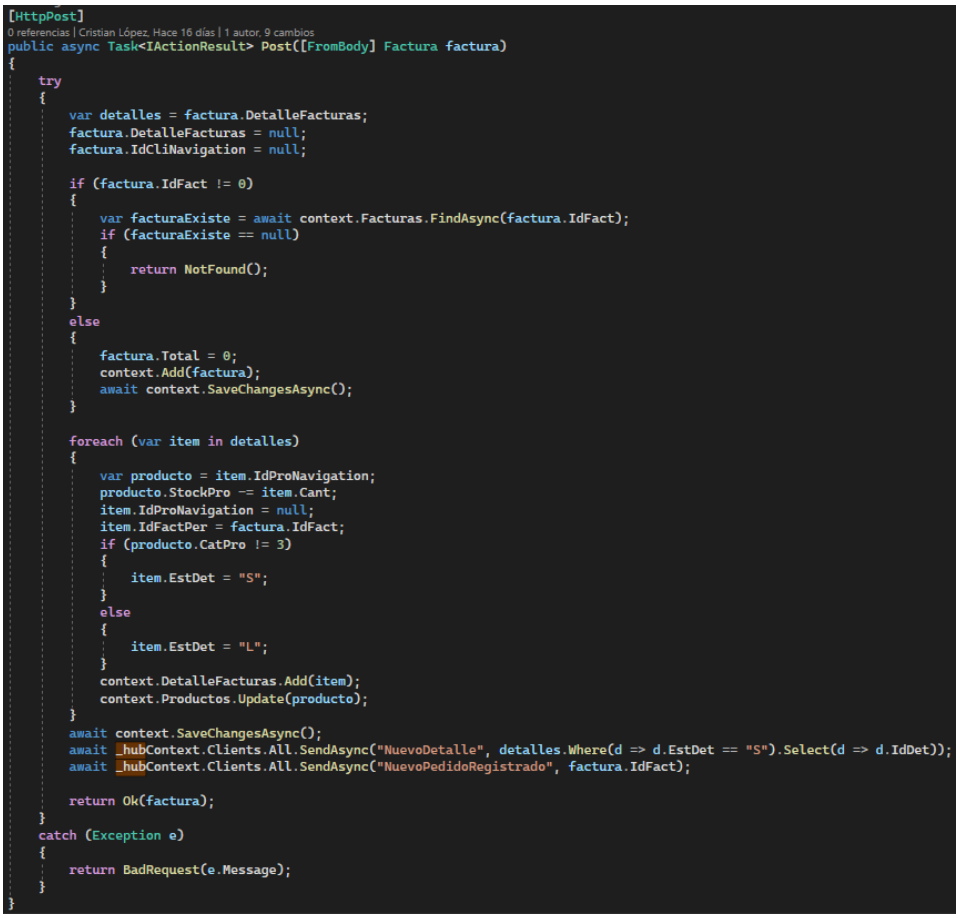
Figura 38. Clase del hub de notificaciones

```
app.UseEndpoints(endpoints =>
{
    endpoints.MapHub<MiHub>("/miHub");
    endpoints.MapControllers();
});
```

Figura 39. Llamada al hub desde el Startup.cs

- ***Módulos – Backend***

Entre las tareas importantes del sistema se encuentran registrar un nuevo pedido, cambiar el estado de los detalles, y cobrar el pedido, acciones que disparan eventos en el hub de notificaciones correspondientemente.



```
[HttpPost]
0 referencias | Cristian López, Hace 16 días | 1 autor, 9 cambios
public async Task<IActionResult> Post([FromBody] Factura factura)
{
    try
    {
        var detalles = factura.DetalleFacturas;
        factura.DetalleFacturas = null;
        factura.IdCliNavigation = null;

        if (factura.IdFact != 0)
        {
            var facturaExiste = await context.Facturas.FindAsync(factura.IdFact);
            if (facturaExiste == null)
            {
                return NotFound();
            }
        }
        else
        {
            factura.Total = 0;
            context.Add(factura);
            await context.SaveChangesAsync();
        }

        foreach (var item in detalles)
        {
            var producto = item.IdProNavigation;
            producto.StockPro -= item.Cant;
            item.IdProNavigation = null;
            item.IdFactPer = factura.IdFact;
            if (producto.CatPro != 3)
            {
                item.EstDet = "S";
            }
            else
            {
                item.EstDet = "L";
            }
            context.DetalleFacturas.Add(item);
            context.Productos.Update(producto);
        }
        await context.SaveChangesAsync();
        await _hubContext.Clients.All.SendAsync("NuevoDetalle", detalles.Where(d => d.EstDet == "S").Select(d => d.IdDet));
        await _hubContext.Clients.All.SendAsync("NuevoPedidoRegistrado", factura.IdFact);

        return Ok(factura);
    }
    catch (Exception e)
    {
        return BadRequest(e.Message);
    }
}
```

Figura 40. Función para registrar nuevo pedido.

```
[HttpPut]
0 referencias | Cristian López, Hace 121 días | 1 autor, 5 cambios
public async Task<IActionResult> PutEstadoOrden(int idDet, string estado)
{
    DetalleFactura detalle;
    try
    {
        detalle = context.DetalleFacturas.Find(idDet);
        detalle.EstDet = estado;
        context.Entry(detalle).State = Microsoft.EntityFrameworkCore.EntityState.Modified;
        await context.SaveChangesAsync();
        if (estado == "L")
        {
            await _hubContext.Clients.All.SendAsync("PedidoListo", idDet);

        }
        else
        {
            if (estado == "E")
            {
                await _hubContext.Clients.All.SendAsync("PedidoPorCobrar", idDet);
            }
            else
            {
                if (estado != "F")
                {
                    await _hubContext.Clients.All.SendAsync("ActualizarEstadoPedido", idDet);
                }
            }
        }

        return Ok(new { message = "Estado actualizado con éxito" });
    }
    catch (Exception e)
    {
        return BadRequest(e.Message);
    }
}
```

Figura 41. Función para cambiar estado del pedido

```
[HttpPut]
[Route("cobrar")]
0 referencias | Cristian López, Hace 121 días | 1 autor, 2 cambios
public async Task<IActionResult> PutTotalActualizado(int idFact, decimal total)
{
    Factura factura;
    try
    {
        factura = context.Facturas.Find(idFact);
        factura.Total += total;
        context.Entry(factura).State = Microsoft.EntityFrameworkCore.EntityState.Modified;
        await context.SaveChangesAsync();
        await _hubContext.Clients.All.SendAsync("PedidoCobrado", idFact);
        return Ok(new { message = "Total actualizado con éxito" });
    }
    catch (Exception e)
    {
        return BadRequest(e.Message);
    }
}
```

Figura 42. Función para cobrar el pedido

- ***Módulos – Frontend***

Las vistas se estructuran acorde a lo definido por el cliente en los mockups, de esta forma se esquematiza como se comunican y operan los diferentes módulos del sistema, y al ser un reflejo de las operaciones en el backend, son las tareas de registro, cambio de estado y cobro de pedidos aquellos que denotan el hecho del negocio.

```
guardarFactura() {
  for (let data of this.itemsCarrito) {···
  }

  if (this.idPedido == null || this.idPedido == "") {···
  }
  const factura: Factura = {···
  }

  const facturaModelo: Factura = {···
  }
  this._facturaService.postFactura(factura).subscribe(data => {
    this.makeText("Pedido realizado con éxito");
    localStorage.removeItem('carrito');

    this.salir();
    this.cargarCarrito();
    this._adminService.postFacturaModelo(facturaModelo).subscribe(data => {
      console.log(data)
    })
  }, error => {
    if (error.status === 404) {
      // El error fue un NotFound()
      this.makeText("El pedido ingresado no existe");
    } else {
      // Otro tipo de error
      this.makeText("Lo siento, hubo un error en la transacción.");
    }
    this.listaDetalle = []
    this.cargarCarrito()
  });
}
salir() {
  this.router.navigate(['/dashboard/compras']);
}
}
```

Figura 43. Función para registrar un nuevo pedido

```
cambiarEstado(element: any) {
  let estado = "";
  let quitar = false;
  if (element.estado == false && element.idProNavigation.catProNavigation == false) {
    estado = "S";
  }
  if (element.estado == true && element.idProNavigation.catProNavigation == false) {
    estado = "P";
  }
  if (element.estado == true && element.idProNavigation.catProNavigation == true) {
    estado = "L";
    quitar = true;
  }

  this._facturaService.actualizarEstadoPedido(element.idDet, estado).pipe(···
  ).subscribe(data => {
    // Encuentra el índice del elemento en la lista 'values' de cada ítem de la estructura
    let pedidoEliminado;
    for (let i = 0; i < this.listaPedidosOptimizados.length; i++) {···
    }
    this.listaRecuentoPedidos = this.agruparYContarPorNombre(this.listaPedidosOptimizados.reduce((acc, curr) => {
      return acc.concat(curr.values);
    }, []));
  });
}
```

Figura 44. Función para cambiar estado del pedido

```
pagar() {
  var detalles = []
  var ventas = JSON.parse(localStorage.getItem('ventas')) || []
  if (this.separados.length === 0) {···
  } else {···
  }
  // Guardar la venta en localStorage
  const padre = {···
  };
  const indice = ventas.findIndex(item => item.idFact === padre.idFact);
  if (indice !== -1) {···
  } else {···
  }
  localStorage.setItem('ventas', JSON.stringify(ventas));//actualizar cookies
}
```

Figura 45. Función para cobrar el pedido

- ***Consumo de las APIs***

Para establecer comunicación entre el frontend y el backend, es necesario consumir las APIs definidas en los controladores por un estándar o protocolo adecuado, en este caso, a través de HTTPs, por lo que la generación de servicios es la mejor alternativa para hacer solicitudes al servidor.

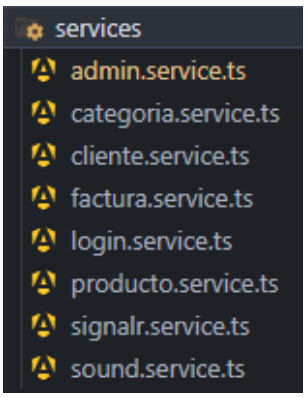


Figura 46. Servicios frontend

- ***Despliegue a Producción***

Con la parte funcional del sistema probado y funcionando, se realizó la implantación en el negocio, para empezar con la fase de recolección de datos para alimentar al modelo de predicción. Se utilizó los servicios de Microsoft Azure como Platform as a Service (PaaS) para alojar la base de datos y los servicios Web, mientras que la aplicación cliente se compiló desde Ionic usando Capacitor para generar la APK, misma que será instalada en los dispositivos Android del personal, y se aloja en el repositorio del proyecto. Ver anexo B.

d. Sprint 4

Con los datos generados por el uso de la aplicación, se pudo extraer las características que mejor representan al hecho del negocio, es decir, las ventas, pasando por la obtención de datos, preprocesamiento y postprocesamiento. Así se puede establecer que se pretende estimar o predecir, generando el modelo para tener los archivos y rutinas necesarias, para posterior a ello acoplarlo al servidor que se encargará de alojar y ejecutar el algoritmo.

- ***Obtención de datos***

Es el módulo en el que se genera el dataset u origen de datos con todas las observaciones del hecho, para tener una base en la cual se pueda empezar a estructurar el modelo, representando mayormente a datos históricos del negocio.

```
try:
    conn = pyodbc.connect(conn_str)
    print('Conexión exitosa')

    # Aquí puedes ejecutar tus consultas y operaciones en la base de datos
    query = """
    SELECT
        a.id_det AS 'ID Detalle',
        a.id_fact_per AS 'ID Pedido',
        d.FEC_FACT AS 'Fecha-Hora',
        b.nom_pro AS 'Producto',
        e.id_cat AS 'Categoria',
        a.est_det AS 'Estado',
        c.NOM_CLI AS 'Mesero Encargado',
        d.ID_MESA AS 'Mesa o Cliente',
        a.CANT AS 'Cantidad',
        a.PU AS 'Precio',
        a.CANT * a.PU AS 'Subtotal',
        d.TOTAL AS 'Total'
    FROM
        Detalle_Factura AS a,
        Productos AS b,
        Clientes AS c,
        Facturas AS d,
        Categorias AS e
    WHERE
        a.ID_PRO = b.ID_PRO AND
        c.ID_CLI = d.ID_CLI AND
        a.ID_FACT_PER = d.ID_FACT AND
        b.CAT_PRO = e.ID_CAT
    ORDER BY
        a.ID_FACT_PER;
"""
    conn = pyodbc.connect(conn_str)
    df = pd.read_sql(query, conn)
    conn.close()
    # Exportar el DataFrame a un archivo CSV
    nombre_archivo = "datos_exportados.csv"
    df.to_csv(nombre_archivo, sep=',', decimal='.', index=False, float_format='%.2f')
    # No olvides cerrar la conexión cuando hayas terminado
except pyodbc.Error as ex:
    print('Error de conexión:', ex)
```

Figura 47. Obtención y creación del dataset

- ***Preprocesamiento***

A grandes rasgos, la información del dataset por si sola se puede analizar para establecer ciertas tendencias o patrones de forma manual si cabe el término, sin embargo, para poder aplicar algún algoritmo de aprendizaje automático, se requieren aplicar ciertas técnicas para etiquetar y escalar los datos, de modo que sea comprensible y manejable para el algoritmo, por todas las condiciones y restricciones que estos poseen. Además de seleccionar las características que mejor definan las observaciones.

```
# Carga el conjunto de datos
data = pd.read_csv('datos_exportados.csv')
dia_a_numero = {
    'Monday': 0,
    'Tuesday': 1,
    'Wednesday': 2,
    'Thursday': 3,
    'Friday': 4,
    'Saturday': 5,
    'Sunday': 6
}
# Convertir 'Fecha-Hora' a datetime y extraer mes, día y hora
data['Fecha-Hora'] = pd.to_datetime(data['Fecha-Hora'])
data['Mes'] = data['Fecha-Hora'].dt.month
data['Día'] = data['Fecha-Hora'].dt.day
data['Hora'] = data['Fecha-Hora'].dt.hour
# Extraer el día de la semana y asignarlo a una nueva columna
data['Día de la Semana'] = data['Fecha-Hora'].dt.day_name()
data['Día de la Semana'] = data['Día de la Semana'].map(dia_a_numero)

# Eliminar las columnas innecesarias
data.drop(['ID Detalle', 'ID Pedido', 'Estado', 'Subtotal','Fecha-Hora','Total'], axis=1, inplace=True)

# Inicializar los codificadores de etiquetas para cada columna categórica
label_encoders = {}
categorical_features = ['Producto', 'Mesero Encargado', 'Mesa o Cliente']
for column in categorical_features:
    label_encoders[column] = LabelEncoder()
    data[column] = label_encoders[column].fit_transform(data[column])

# Guardar los codificadores de etiquetas para su uso futuro en predicciones
joblib.dump(label_encoders, 'label_encoders.joblib')

# Estandarizar los datos (importante para muchos algoritmos de clustering)
scaler = StandardScaler()
scaled_data = scaler.fit_transform(data)

# Guardar el escalador
joblib.dump(scaler, 'scaler.joblib')
```

Figura 48. Tratamiento y procesamiento de datos

- ***Postprocesamiento***

La etapa en la que se genera un dataset con datos para el aprendizaje de máquina, estableciendo la variable objetivo, o, en otras palabras, aquello que el algoritmo que se diseñará va a predecir, que para este caso, al tener una correlación no lineal entre los datos, fue el algoritmo de clusterización KMeans el que fungió como generador de la variable objetivo, buscando patrones entre los datos para agruparlos en 3 clusters, que representan las distintas ventas que se realizan comúnmente en el negocio.

```
for i in range(1, 4):
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10, random_state=0)
    kmeans.fit(scaled_data)


# Etiquetas de los clusters
clusters = kmeans.labels_

# Añadir las etiquetas de los clusters al dataframe original
data['Cluster'] = clusters

# Guardar el resultado en un nuevo archivo CSV
data.to_csv('datosClusterizados.csv', index=False)
```

Figura 49. Generación de la variable objetivo

- ***Red Neuronal***

La red neuronal está pensada para que, hallado los patrones y generados los grupos o clústers de ventas, realice la predicción de nuevos pedidos ingresados, de modo que, conociendo a que grupo pertenece cada pedido realizado, el administrador puede decidir cómo cambiar el rumbo del negocio en el momento; ya sea en la atención, la oferta de productos, las expectativas de retorno, entre otras, todo dentro del marco del servicio brindado diariamente, basándose en los criterios de la Tabla 11.

Tabla 11. Criterios generales para la predicción

Criterios	**Cluster 1**	**Cluster 2**	**Cluster 3**
Frecuencia	Medianamente solicitado	Altamente solicitado	Poco solicitado
Precio	Bajo	Medio o Alto	Bajo o Alto
Cantidad	Bajo	Bajo o Medio	Bajo o Alto
Categoría	Mayormente entradas, bebidas, extras.	Platos Fuertes	Platos Fuertes, cocteles.

El proceso de modelado y construcción de la red partió desde los datos postprocesados, aplicando técnicas de escalado para regularizar los coeficientes, de modo que durante el proceso de entrenamiento y testeo el modelo no tenga mucha variación.

```
#Obtener las dimensiones correctas del dataset
num_columnas = data.shape[1] - 1
num_datos_unicos = data[data.columns[-1]].nunique()
print(num_columnas,num_datos_unicos)

# Separar datos en entradas y etiquetas
X = data.iloc[:, :-1].values  # Todas las filas, todas las columnas excepto la última
y = data.iloc[:, -1].values  # Todas las filas, última columna

# Dividir datos en entrenamiento y validación
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

# Escalar datos
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Figura 50. Escalado y división de datos

La arquitectura de la red neuronal dependerá mucho de distintos factores, como la complejidad de lo que se quiere predecir, la cantidad de características de los datos, la cantidad de datos generales, entre otras.

```
# Crear modelo de red neuronal
model = Sequential()
model.add(Dense(10, input_dim=num_columnas, activation='relu' )) #capa oculta y con una entrada de n campos
model.add(Dense(9, activation='relu'))
model.add(Dense(num_datos_unicos, activation='softmax')) #capa de salida con salida de 3 opciones

# Compilar modelo
model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# Entrenar modelo
history = model.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_test, y_test))
```

Figura 51. Creación de la red neuronal

El código expuesto en la Figura 51 define, compila y entrena un modelo de red neuronal para un problema de clasificación multiclase utilizando el conjunto de datos proporcionado (X_train, y_train, X_test, y_test). La arquitectura de la red consta de una capa de entrada con 10 neuronas, una capa oculta con 9 neuronas, y una capa de salida con un número de neuronas igual a la cantidad de clústers, en este caso, 3. La función de pérdida utilizada es sparse_categorical_crossentropy y el optimizador es 'adam'. La métrica de evaluación durante el entrenamiento es la precisión (accuracy).

Se obtuvo un 99,67% de precisión del modelo, y un 99,71% de precisión durante el entrenamiento, con 10 épocas de entrenamiento.

```
# Calcular la precisión del modelo
test_loss, test_accuracy = model.evaluate(X_test, y_test, verbose=0)
print("Pérdida de prueba: {:.2f}%".format(test_loss))
print("Precisión del modelo (test): {:.2f}%".format(test_accuracy * 100))

# Calcular la precisión de entrenamiento
train_loss, train_accuracy = model.evaluate(X_train, y_train)
print("Pérdida de entrenamiento: {:.2f}%".format(test_loss))
print("Precisión del entrenamiento: {:.2f}%".format(train_accuracy * 100))
```

```
Pérdida de prueba: 0.05%
Precisión del modelo (test): 99.67%
77/77 [==============================] - 0s 832us/step - loss: 0.0434 - accuracy: 0.9971
Pérdida de entrenamiento: 0.05%
Precisión del entrenamiento: 99.71%
```

Figura 52. Resultados de precisión del modelo

Se guarda el modelo para trabajar con este en la construcción del servidor.

```
# Guardar el modelo entrenado
model.save('model.h5')
```

Figura 53. Generación del archivo del modelo

e. Sprint 5

Se cubrió todas las tareas correspondientes a la creación y configuración del servidor del modelo, así como la integración de este con la aplicación móvil, teniendo en cuenta su rendimiento, su precisión y retroalimentación periódica, para consecuentemente, enviar todas estas actualizaciones a producción.

- ***Creación y configuración del servidor Django***

Ya que el modelo fue construido netamente en Python, se requiere de un servidor que pueda ejecutar Python en la web, para hacer peticiones por medio de https y acceder al modelo para realizar predicciones, y Django es un framework que facilita esta tarea, proporcionando herramientas para el despliegue y ejecución de código Python en la web.

En cuanto a las predicciones realizadas y las métricas de estimación de venta, se hizo uso de la base de datos de sqlite3 que se genera junto con el proyecto de Django para poder almacenar esta información, de modo que se lleve un registro y no se deba recalcular todo al momento de realizar nuevas predicciones.

Figura 54. Base de datos SQLite3 propia del proyecto

Se crearon dos modelos, que representan las tablas, una para poder almacenar los valores de la métrica de la agrupación de cada día de la semana, del mes actual, y otra tabla para poder almacenar las predicciones que se van realizando a medida que se usa el software.

```python
from django.db import models

# Create your models here.
class ConteoCluster(models.Model):
    cluster = models.IntegerField(primary_key=True)
    conteo_lunes = models.IntegerField(default=0)
    conteo_martes = models.IntegerField(default=0)
    conteo_miercoles = models.IntegerField(default=0)
    conteo_jueves = models.IntegerField(default=0)
    conteo_viernes = models.IntegerField(default=0)
    conteo_sabado = models.IntegerField(default=0)
    conteo_domingo = models.IntegerField(default=0)

    def __str__(self):
        return f"Cluster {self.cluster}"

class Prediccion(models.Model):
    nomPro = models.CharField(max_length=100)
    catPro = models.IntegerField()
    idCli = models.CharField(max_length=100)
    idMesa = models.IntegerField()
    cantidad = models.IntegerField()
    precio = models.FloatField()
    mes = models.IntegerField()
    dia = models.IntegerField()
    diaSemana = models.IntegerField()
    hora = models.IntegerField()
    categoria_predicha = models.IntegerField()

    def __str__(self):
        return f"{self.nomPro} - {self.categoria_predicha}"
```

Figura 55. Modelos para la base de datos SQLite3

```
ALLOWED_HOSTS = ["*"]


# Application definition

INSTALLED_APPS = [
    'miAPI',
    "django.contrib.admin",
    "django.contrib.auth",
    "django.contrib.contenttypes",
    "django.contrib.sessions",
    "django.contrib.messages",
    "django.contrib.staticfiles",
    "corsheaders",
    'django_cron',
]

CRON_CLASSES = [
    'miAPI.crons.ReentrenamientoCronJob',  # Ruta a tu clase de tarea cron
]

MIDDLEWARE = [
    "django.middleware.security.SecurityMiddleware",
    "django.contrib.sessions.middleware.SessionMiddleware",
    "django.middleware.common.CommonMiddleware",
    "django.middleware.csrf.CsrfViewMiddleware",
    "django.contrib.auth.middleware.AuthenticationMiddleware",
    "django.contrib.messages.middleware.MessageMiddleware",
    "django.middleware.clickjacking.XFrameOptionsMiddleware",
    "corsheaders.middleware.CorsMiddleware",
    'whitenoise.middleware.WhiteNoiseMiddleware'
]

CORS_ALLOW_ALL_ORIGINS = True
```

Figura 56. Configuración en settings.py

En el archivo settings.py se establecen valores importantes para el funcionamiento del proyecto, como las apps instaladas, middlewares, los hosts que podrán hacer peticiones al servidor, entre otros.

```
from django.contrib import admin
from django.urls import path
from miAPI.views import predecir_pedido, obtener_conteo_clusters
from django.conf import settings
from django.conf.urls.static import static
urlpatterns = [
    path("admin/", admin.site.urls),
    path('api/predecir/', predecir_pedido, name='predecir_pedido'),
    path('api/conteoclusters/', obtener_conteo_clusters, name='conteo-clusters'),
] + static(settings.STATIC_URL, document_root=settings.STATIC_ROOT)
```

Figura 57. Rutas para consumo de servicios en urls.py

En el archivo urls.py se definen las rutas que vienen siendo APIs que luego se podrán consumir a través de peticiones https, estas representan las funciones definidas en el archivo views.py.

```python
@csrf_exempt
@require_http_methods(["POST"])
def predecir_pedido(request):
    try:
        # Llamada a los modelos
        label_encoders = joblib.load('miAPI/models/label_encoders.joblib')
        modelo = load_model('miAPI/models/model.h5')
        scaler = joblib.load('miAPI/models/scaler.joblib')

        # Parsear el JSON del cuerpo del request
        data = json.loads(request.body)

        # Extraer detalles de la factura y otros datos relevantes
        detalles_factura = data.get('detalleFacturas', [])
        id_cli = data.get('idCli')
        id_mesa = data.get('idMesa')
        fec_fact = data.get('fecFact')

        # Procesar la fecha para obtener mes, día y hora
        fecha_procesada = datetime.fromisoformat(fec_fact)
        mes = fecha_procesada.month
        dia = fecha_procesada.day
        hora = fecha_procesada.hour
        nombre_dia_semana = fecha_procesada.strftime('%A')
        # Usar el mapeo para obtener el número del día de la semana
        dia_semana_numero = dia_a_numero[nombre_dia_semana]
        # Crear y procesar los vectores de características
        vectores_caracteristicas = []
        for detalle in detalles_factura:
            vector = [
                detalle['idProNavigation']['nomPro'],
                detalle['idProNavigation']['catPro'],
                id_cli,
                id_mesa,
                detalle['cant'],
                detalle['pu'],
                mes,
                dia,
                dia_semana_numero,
                hora
            ]
            vector_procesado = preprocesar_detalle(vector, label_encoders)
            if vector_procesado is not None:
                vectores_caracteristicas.append(vector_procesado)

        # Verificar si más del 50% de los vectores fueron excluidos
        if len(vectores_caracteristicas) < len(detalles_factura) / 2:
            return JsonResponse({'mensaje': 'Insuficiente cantidad de datos

        # Escalar los vectores de características
        vectores_escalados = scaler.transform(vectores_caracteristicas)

        # Realizar la predicción y obtener la categoría para cada vector
        categorias_predichas = []
        for indice,vector in enumerate(vectores_escalados):
            vector_reshaped = np.array([vector])  # Convierte a un array 2D
            pred = modelo.predict(vector_reshaped)
            categoria_predicha = np.argmax(pred)
            # Convertir a int nativo de Python si es necesario
            if isinstance(categoria_predicha, np.int64):
                categoria_predicha = int(categoria_predicha)
            categorias_predichas.append(categoria_predicha)

            # Crear y guardar el objeto Prediccion en la base de datos
            Prediccion.objects.create(
                nomPro=vectores_caracteristicas[indice][0],
                catPro=vectores_caracteristicas[indice][1],
                idCli=vectores_caracteristicas[indice][2],
                idMesa=vectores_caracteristicas[indice][3],
                cantidad=vectores_caracteristicas[indice][4],
                precio=vectores_caracteristicas[indice][5],
                mes=vectores_caracteristicas[indice][6],
                dia=vectores_caracteristicas[indice][7],
                diaSemana=vectores_caracteristicas[indice][8],
                hora=vectores_caracteristicas[indice][9],
                categoria_predicha=categoria_predicha
            )
        # Envía la respuesta
        return JsonResponse({'predicciones': categorias_predichas})

    except Exception as e:
        return JsonResponse({'error': str(e)})
```

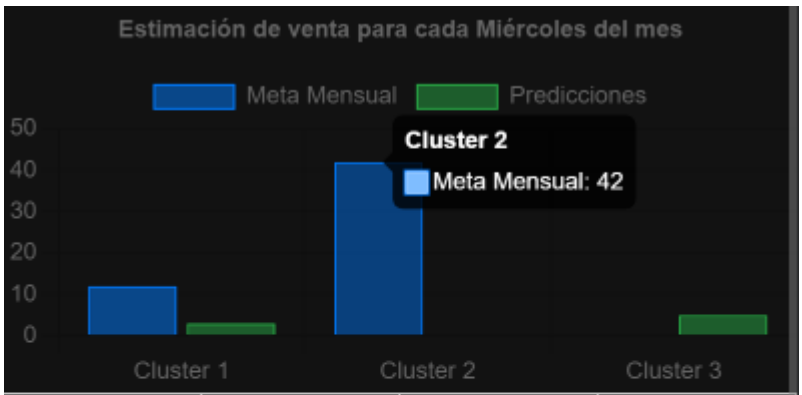
Figura 58. APIs para la predicción usando el modelo

- ***Integración con la aplicación móvil***

Al tratarse de un backend para el consumo de los servicios del modelo, se creó un servicio en el frontend con la URL del servidor, definiendo las APIs de consumo, y llamándolas en donde corresponde al momento de registrar una nueva orden.

```typescript
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { Observable } from 'rxjs';
import { Factura } from '../interfaces/factura';


@Injectable({
  providedIn: 'root'
})
export class AdminService {
  private apiUrl = 'https://api-modelo-ramones.fly.dev/api/';

  constructor(private http: HttpClient) { }

  getConteoClusters(fecha: any): Observable<any> {
    return this.http.get(this.apiUrl+'conteoclusters/'+ "?fecha="+fecha, fecha);
  }

  postFacturaModelo(factura: Factura): Observable<any> {
    return this.http.post(this.apiUrl+'predecir/', factura);
  }
}
```

Figura 59. Servicio para consumo de APIs del modelo



Criterio	Cluster 1	Cluster 2	Cluster 3
Frecuencia	Medianamente solicitado	Altamente solicitado	Poco solicitado
Precio	Bajo	Medio o Alto	Bajo o Alto
Cantidad	Bajo	Bajo o Medio	Bajo o Alto
Categoría	Mayormente entradas, extras.	Platos Fuertes	Platos Fuertes, cocteles.

Figura 60. Métricas de estimación por clúster (App Móvil)

- ***Reentrenamiento***

A medida que la aplicación se sigue utilizando, se siguen generando datos, y esto implica que el modelo debe nutrirse a la par, por ello se realiza el reentrenamiento de este, pasando por la extracción, preprocesamiento y postprocesamiento, todo por una rutina definida que se ejecutará cada domingo siempre y cuando se sobrepase cierta cantidad de datos nuevos.

```python
from django_cron import CronJobBase, Schedule
import subprocess
import os
from datetime import datetime

class ReentrenamientoCronJob(CronJobBase):
    RUN_AT_TIMES = ['23:59']  # Ejecutar a las 23:59
    schedule = Schedule(run_at_times=RUN_AT_TIMES)
    code = 'miAPI.crons.ReentrenamientoCronJob'  # Un identificador único para este job

    def do(self):
        today = datetime.now()
        if today.weekday() == 6:  # 0 es lunes, 6 es domingo
            project_path = os.path.abspath(os.path.dirname(__name__))
            script_path = os.path.join(project_path, 'Reentrenamiento.py')

            # Ejecuta el script
            subprocess.run(['python', script_path], check=True)
```

Figura 61. Rutina para reentrenar el modelo

Es justamente en el proceso de reentrenamiento donde se actualizan las métricas en la base de datos, en base a nuevos datos generados, de este modo, el modelo se acopla a los cambios ocurridos en las ventas, es decir, si se han aumentado las ventas, los KPIs serán más elevados y por ende metas más ambiciosas por cumplir.

```python
# Agrupamos por 'Cluster', 'Día de la Semana' y 'Mes' y contamos las ocurrencias
conteos = data.groupby(['Cluster', 'Día de la Semana', 'Mes']).size()

# Resetear el índice para poder hacer cálculos en las columnas
conteos_reset = conteos.reset_index(name='Conteo')

# Agrupar por 'Cluster' y 'Día de la Semana' y calcular la media para cada combinación a través de los meses
conteos_media = conteos_reset.groupby(['Cluster', 'Día de la Semana'])['Conteo'].mean()

# Desagrupamos para convertir 'Día de la Semana' en columnas y los clusters en filas
conteos_por_dia = conteos_media.unstack(fill_value=0)

# Iterar sobre el DataFrame y actualizar la base de datos
for cluster_id in conteos_por_dia.index:
    ConteoCluster.objects.update_or_create(
        cluster=cluster_id,
        defaults={
            'conteo_lunes': conteos_por_dia.loc[cluster_id, 0],
            'conteo_martes': conteos_por_dia.loc[cluster_id, 1],
            'conteo_miercoles': conteos_por_dia.loc[cluster_id, 2],
            'conteo_jueves': conteos_por_dia.loc[cluster_id, 3],
            'conteo_viernes': conteos_por_dia.loc[cluster_id, 4],
            'conteo_sabado': conteos_por_dia.loc[cluster_id, 5],
            'conteo_domingo': conteos_por_dia.loc[cluster_id, 6],
        }
    )
```

Figura 62. Actualización de métricas

- ***Prueba y despliegue***

Se testeó la eficiencia del modelo para obtener predicciones de los pedidos, en cuanto a cómo los clasificaba en cada clúster, una vez se constató que las predicciones eran correctas y acertadas, se lo liberó para ser utilizado de manera permanente.

A lo largo del desarrollo del proyecto, se iba midiendo la satisfacción de los usuarios en las reuniones scrum, con la finalidad de cumplir con los requerimientos que estos planteaban en cada reunión, basándose en características propias de cualquier software.

Tabla 12. Satisfacción de usuarios

Características del Software	**Percepción por mayoría de usuarios**				
	Reunión 1	**Reunión 2**	**Reunión 3**	**Reunión 4**	**Reunión 5**
Usable	No	No	Si	Si	Si
Eficiente	No	Si	Si	Si	Si
Sencilla	No	No	No	Si	Si
Retroalimentación	No	No	No	No	Si
Segura	Si	Si	Si	Si	Si

Con la solución implantada, se pudieron notar varios cambios por parte del personal, ellos aseveran que las confusiones en los pedidos al momento del cobro han disminuido significativamente, los tiempos de atención al cliente son menores, los procesos de gestión se han optimizado, el esfuerzo por retención de información es nulo, los recursos en cocina no se desperdician, las estimaciones aportan a planificar la atención del día, y además, por todo lo que implica estos cambios, la rentabilidad del negocio incrementó en un 24,7% desde que se empezó a utilizar la solución, esto se ha podido constatar por los registros contables del negocio, que por motivos de confidencialidad no pueden ser presentados en esta investigación.

3.2.5 Fase 5: Documentación

La correcta utilización de la aplicación móvil depende mucho de buenas prácticas de diseño y conocimientos básicos de interacción hombre-máquina, sin embargo, resulta crucial brindarle al usuario una guía de cómo funciona la herramienta, y que puede hacer esta por él, para eso se creó el manual de usuario de “Ramones Manager”, que se encuentra en el ANEXO A.

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

Del presente proyecto de investigación se puede concluir que:

- Se evidenciaron cuellos de botella en los procesos que realizaba la empresa, arraigado a la tradicionalidad, por lo que un 95% estuvo de acuerdo con implantar una solución que atendiera a las distintas problemáticas existentes en las múltiples áreas de gestión, siendo precisamente el uso de tecnologías el medio principal de atención a estos requerimientos, de la mano de un rediseño de los procesos.

- La elección de un determinado modelo de predicción dependerá en gran medida de los objetivos del proyecto, y lo que se quiera lograr a corto y largo plazo con el algoritmo, ya que si bien existen variados modelos cuya implementación se simplifica gracias a librerías completas existentes en Python, no todos pueden ser aplicables a todas las áreas y estructuras empresariales.

- La elección de frameworks adecuados para la ejecución de un proyecto de desarrollo de software que facilitan el proceso de desarrollo en distintas etapas, y una buena elección de metodología ágil que se adapte al cliente, al equipo y a los recursos disponibles, asegurando flexibilidad ante imprevistos, son opciones que aportan al cumplimiento de los objetivos de un proyecto.

- En conclusión, se puede afirmar que la implantación de tecnologías que usan modelos de predicción es posible en pequeñas y medianas empresas, independientemente del tipo de negocio que se trate, se requerirá destinar mayor o menor inversión, sin perder de vista que diseñar un buen plan de ejecución determinará en gran medida el éxito del proyecto.

4.2 Recomendaciones

- Se recomienda, a nivel general, adoptar tecnologías modernas y realizar un análisis de los procesos a fin de determinar si requieren de un rediseño o ajuste para optimizar las operaciones de cualquier PYME, involucrando a los equipos clave de las distintas áreas de gestión.

- Se sugiere ir por soluciones que brinden suficiente retroalimentación al negocio teniendo una arquitectura adaptable y escalable, como lo son las redes neuronales artificiales. Además, sería beneficioso explorar la integración de diferentes algoritmos para crear un enfoque híbrido que pueda adaptarse mejor a las complejidades específicas de la empresa.

- Se sugiere para Ramones Restaurant implementar un plan de seguimiento tras la introducción de la aplicación móvil. Este plan debería enfocarse en evaluar periódicamente cómo funcionan los modelos de predicción y recoger opiniones de usuarios y empleados para realizar mejoras continuas en la aplicación. Es esencial que la empresa se mantenga flexible para actualizar la tecnología según las necesidades del negocio y las expectativas de los clientes.

- Se recomienda fomentar investigaciones futuras que exploren el impacto de la implementación de nuevas tecnologías en la gestión de negocios, especialmente en pequeñas y medianas empresas. Sería relevante investigar cómo estas tecnologías pueden adaptarse a diferentes tipos de negocios y cuáles son las mejores prácticas para su implementación.

REFERENCIAS BIBLIOGRÁFICAS

[1] M. P. Espinosa-Vélez y V. A. Armijos-Buitrón, “La transformación digital y su incidencia en el e-commerce en Ecuador”, en CICIC 2022 - Decima Segunda Conferencia Iberoamericana de Complejidad, Informatica y Cibernetica en el contexto de the 13th International Multi-Conference on Complexity, Informatics, and Cybernetics, IMCIC 2022 - Memorias, International Institute of Informatics and Cybernetics, IIIC, 2021, pp. 169–174. doi: 10.54808/CICIC2022.01.169.

[2] M. E. López Duque, L. E. Restrepo de Ocampo, y G. L. López Velásquez, “Opposition to change in Modern Organizations”, Scientia et Technica Año XVIII, vol. 18, núm. 1, 2013.

[3] C. A. Romero Romero, “Estudio Comparativo De Algoritmos De Inteligencia Artificial Y Mineria De Datos Enfocados a La Toma De Decisiones Empresariales De Seleccion De Personal”, Universidad De Cundinamarca. pp. 3–5, 2018.

[4] S. L. Pazo y P. D. Toaquiza, “DESARROLLO DE UNA APLICACIÓN MÓVIL PARA MONITOREO, CONTROL DE LAS SOLICITUDES Y ENTREGAS DE PEDIDOS MEDIANTE GEOLOCALIZACIÓN EN TIEMPO REAL EN LA EMPRESA AKI VOY DEL CANTÓN LA MANÁ.”, La Maná.

[5] M. R. Zambrano y E. H. Buenaño Valencia, “DESARROLLO DE UNA TIENDA VIRTUAL PROGRESSIVE WEB APPS (PWA) PARA GESTIONAR LAS VENTAS DE LOS PRODUCTOS EN LA EMPRESA GARCÉS TORRES (GT) JEAN’S CUPIDO”, pp. 6–15, 2021.

[6] M. Villanueva y B. Collado López, “Desarrollo de una aplicación móvil para la gestión de pedidos”, Universitat Jaume, Castellón de la Plana, 2022.

[7] E. L. Martínez, “La calidad en el servicio y el nivel de satisfacción del cliente”, Universidad Técnica de Ambato, Ambato, 2021.

[8] A. Tomas, L. Vera, Q. Noemí, S. Vera, J. Xavier, y P. Ganchozo, “Aplicación móvil de venta y entrega de productos del Supermarket Supercito de la ciudad de Calceta, Ecuador”, Mikarimin, pp. 1–4, sep. 2021.

[9] M. M. Acosta-Véliz y M. E. Jiménez-Cercado, “Modelo de gestión empresarial del Ecuador”, Revista Científica FIPCAEC (Fomento de la investigación y publicación científico-técnica multidisciplinaria). ISSN : 2588-090X . Polo de Capacitación, Investigación y Publicación (POCAIP), vol. 5, núm. 5, 2020.

[10] P. Gutiérrez Falcón, “Sistemas de gestión en micro y pequeñas empresas. Metodología para su implementación”, Revista Venezolana de Gerencia, vol. 27, núm. Edición Especial 7, 2022, doi: 10.52080/rvgluz.27.7.41.

[11] M. Torres, I. Barrientos Núñez, y J. C. Quintana Zaez, “Módulo del sistema informático gestión de incidencias para la toma de decisiones”, Universidad de Ciego de Ávila Máximo Gómez Báez, vol. 14, núm. 5, 2021.

[12] E. O. Corzo Durand, “Desarrollo de una Aplicación Web Progresiva (PWA) basado en el framework Laravel para la Gestión de Pedidos en el Proceso de Delivery”, 2021.

[13] E. Córdova, “El servicio al cliente y la competitividad en los restaurantes del cantón Ambato”. Consultado: el 9 de noviembre de 2023. [En línea]. Disponible en: https://repositorio.uta.edu.ec/bitstream/123456789/31481/1/619%20OE.pdf

[14] J. J. Mesias, “APLICACIÓN PARA LA GESTIÓN DE ÓRDENES EN RESTAURANTES DE LA CIUDAD DE AMBATO UTILIZANDO TECNOLOGÍA MÓVIL”, Consultado: el 9 de noviembre de 2023. [En línea]. Disponible en: https://repositorio.uta.edu.ec/bitstream/123456789/31306/1/t1710si.pdf

[15] M. del R. Rodríguez-Cubillo, H. Del Castillo, y B. Arteaga Martínez, “El uso de aplicaciones móviles en el aprendizaje de las matemáticas: una

revisión sistemática", ENSAYOS. Revista de la Facultad de Educación de Albacete, vol. 1, núm. 36, 2021, doi: 10.18239/ensayos.v36i1.2631.

[16] J. Castellanos Claramunt, "La gestión de la información en el paradigma algorítmico: inteligencia artificial y protección de datos", Métodos de informacion, vol. 11, núm. 21, pp. 059–082, 2020, doi: 10.5557/iimei11-n21-059082.

[17] O. Tinoco Gómez, P. P. Rosales López, y J. Salas Bacalla, "Criterios de selección de metodologías de desarrollo de software", Industrial Data, vol. 13, núm. 2, 2014, doi: 10.15381/idata.v13i2.6191.

[18] L. M. Vargas Ordoñez, L. F. Muñoz Sanabria, y F. J. Álvarez, "ALEDO Algoritmo para la detección de objetos", Tecnología Educativa Revista CONAIC, vol. 6, núm. 1, 2021, doi: 10.32671/terc.v6i1.52.

[19] Cristóbal Alejandro Rodríguez Vásconez, "Aplicación de algoritmos de Machine Learning para predecir la deserción estudiantil en alumnos de primer y segundo semestre en universidades públicas del Ecuador." Consultado: el 9 de noviembre de 2023. [En línea]. Disponible en: https://repositorio.uta.edu.ec/bitstream/123456789/38615/1/t2303mma.pdf

[20] I. Mayra Cristina Llumitasig Galarza Director y I. Fabián Rodrigo Salazar Escobar, "SIMULACIÓN DE PRONÓSTICOS DE VENTAS EN LA EMPRESA IMPACTEX MEDIANTE REDES NEURONALES". Consultado: el 9 de noviembre de 2023. [En línea]. Disponible en: https://repositorio.uta.edu.ec/bitstream/123456789/33778/1/t1892mma.pdf

ANEXOS

Anexo A. Manual de Usuario

En la Figura A1 se muestra la portada del manual de usuario.

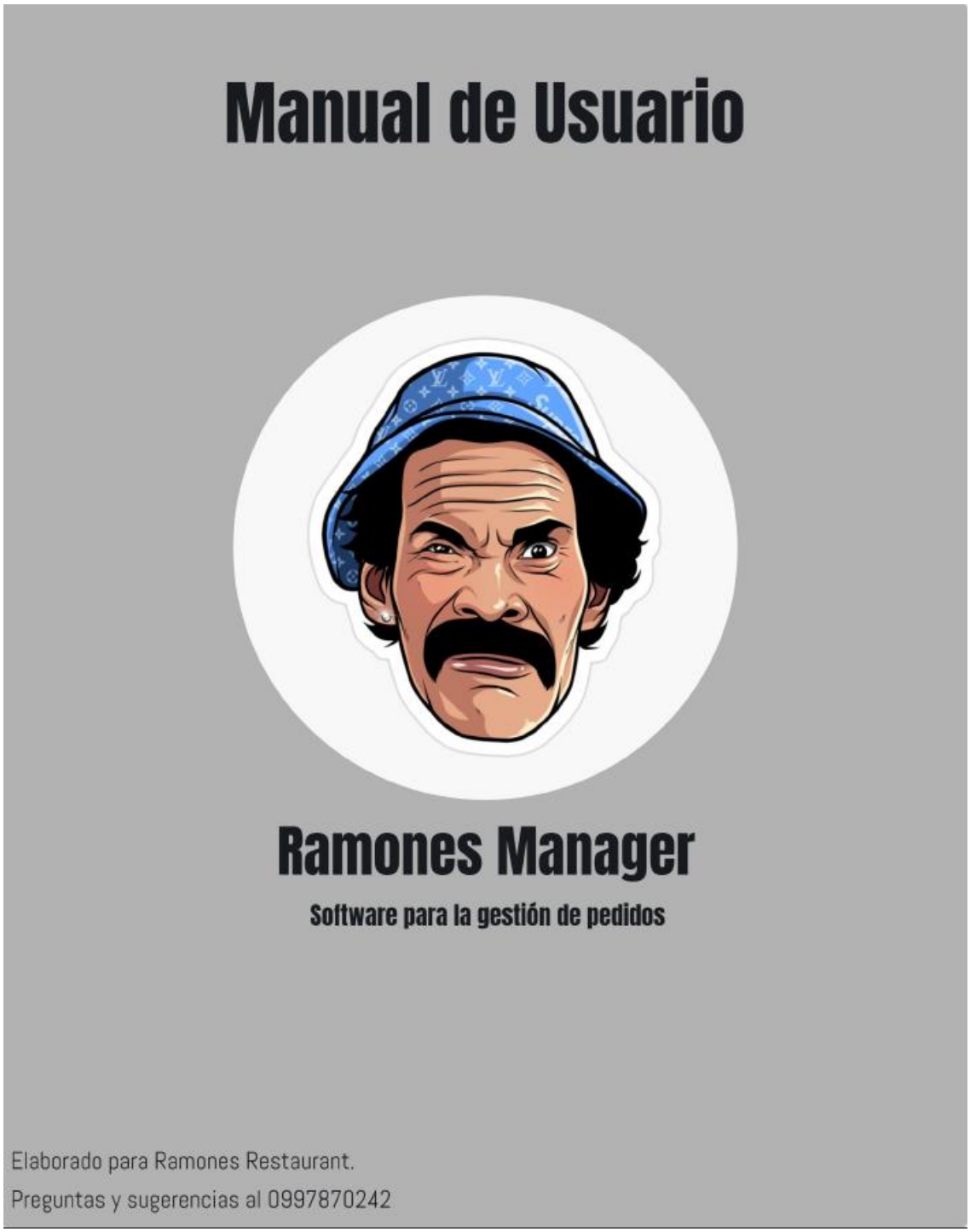


Figura A1. Portada

En la Figura A2 se muestra el módulo de Login.

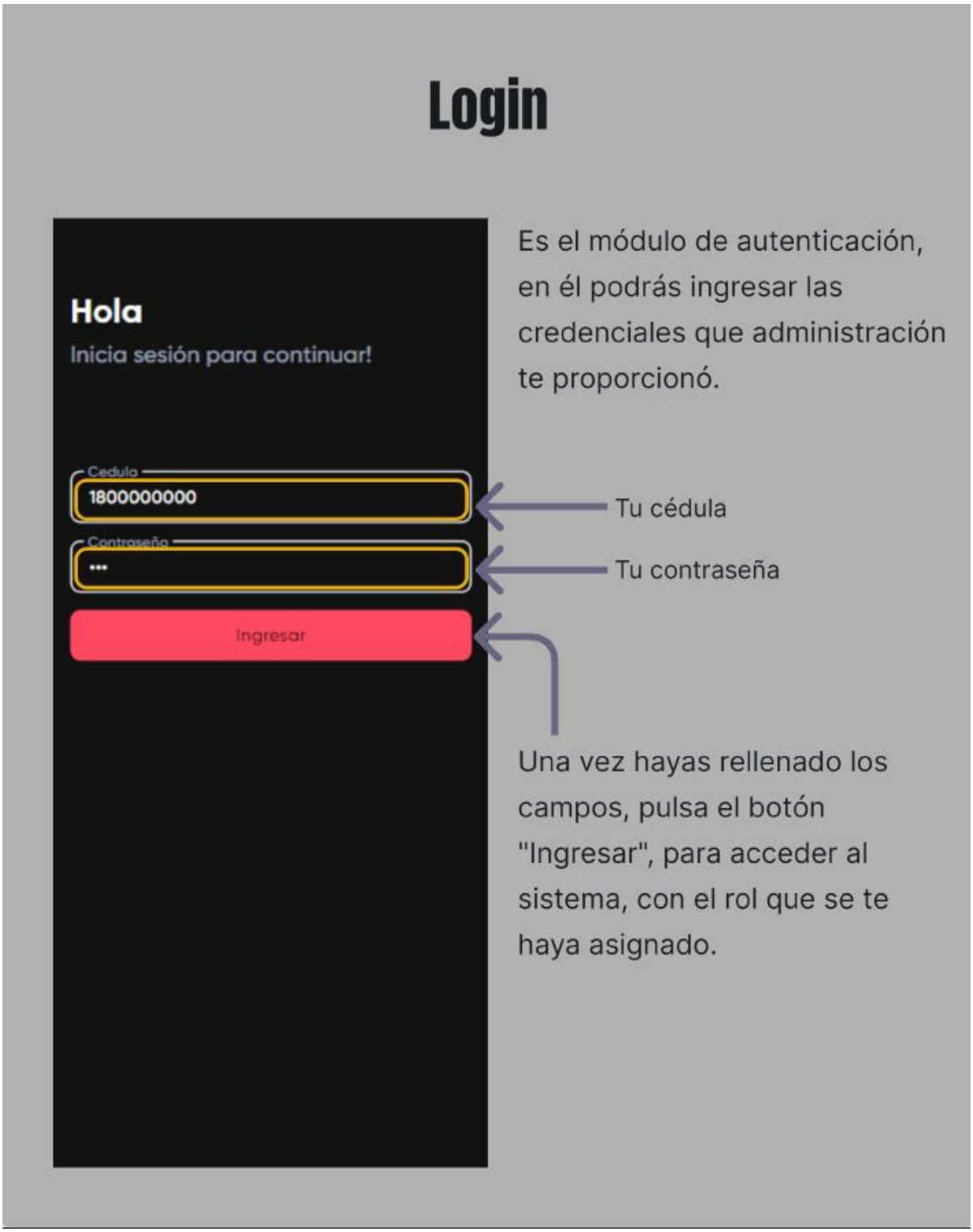


Figura A2. Módulo Login

En la Figura A3 se muestra el módulo del Perfil.

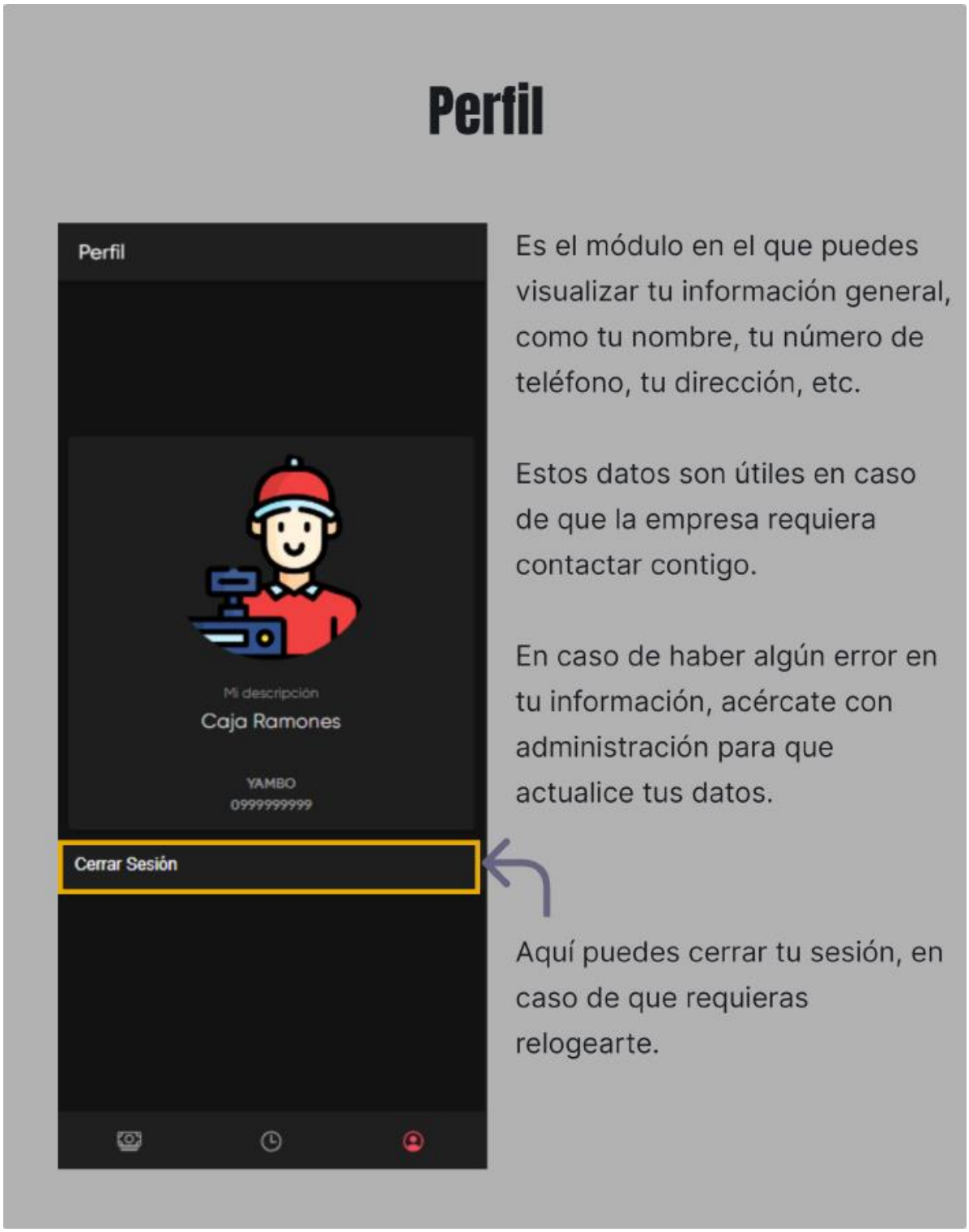


Figura A3. Módulo Perfil

En la Figura A4 se muestra la introducción al módulo de Mesero.

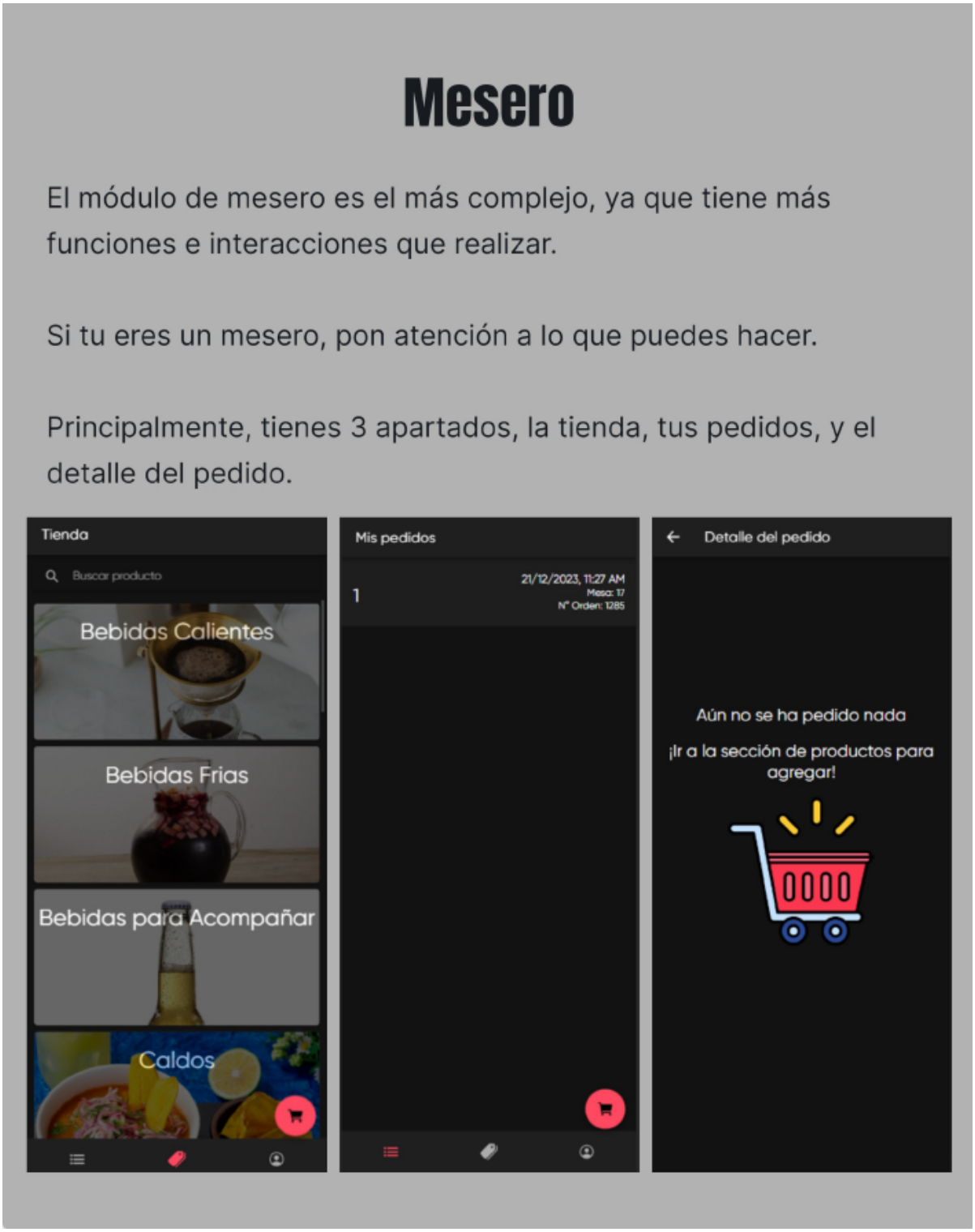


Figura A4. Módulo Mesero

En la Figura A5 se muestra el módulo de Tienda para el Mesero.

Figura A5. Módulo Tienda

En la Figura A6 se muestra el módulo de Productos de la Tienda.

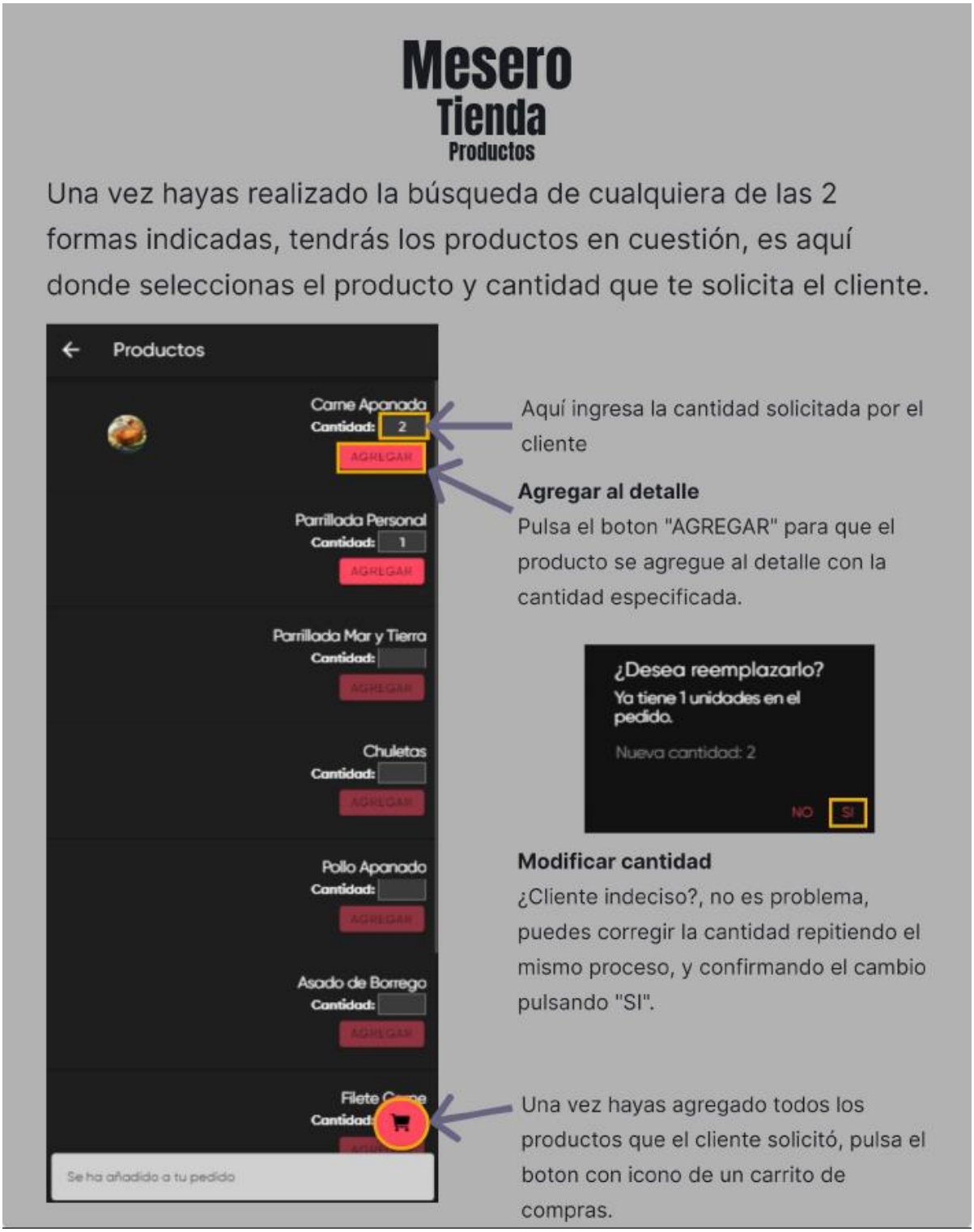


Figura A6. Módulo Productos de la Tienda

En la Figura A7 se muestra el módulo del Detalle del Pedido.

Figura A7. Módulo Detalle del Pedido

En la Figura A8 se muestra el módulo de Pedidos del Mesero.

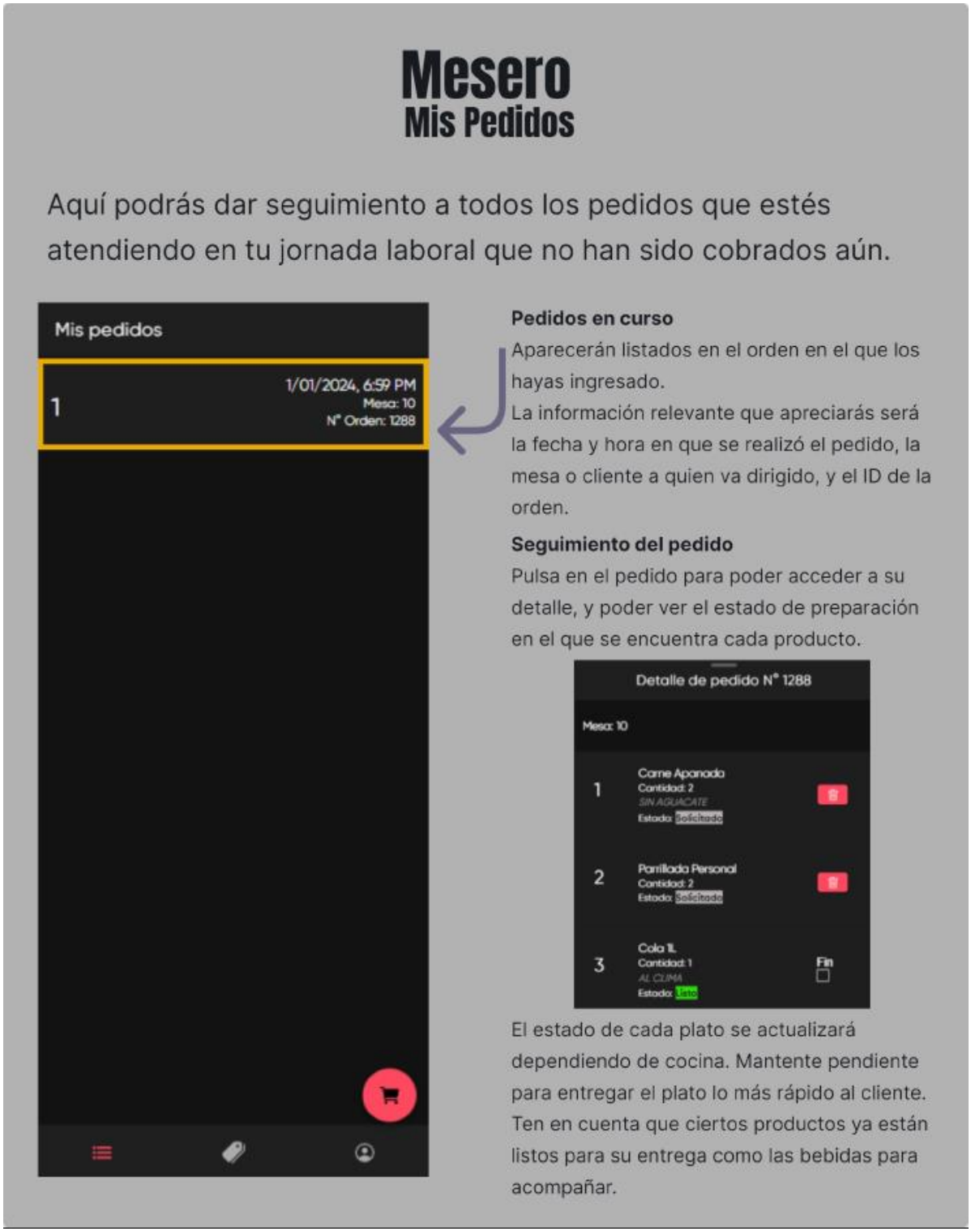


Figura A8. Módulo Pedidos del Mesero

En la Figura A9 se muestra el módulo de Cocina.

Figura A9. Módulo Cocina

En la Figura A10 se muestra el módulo de Caja.

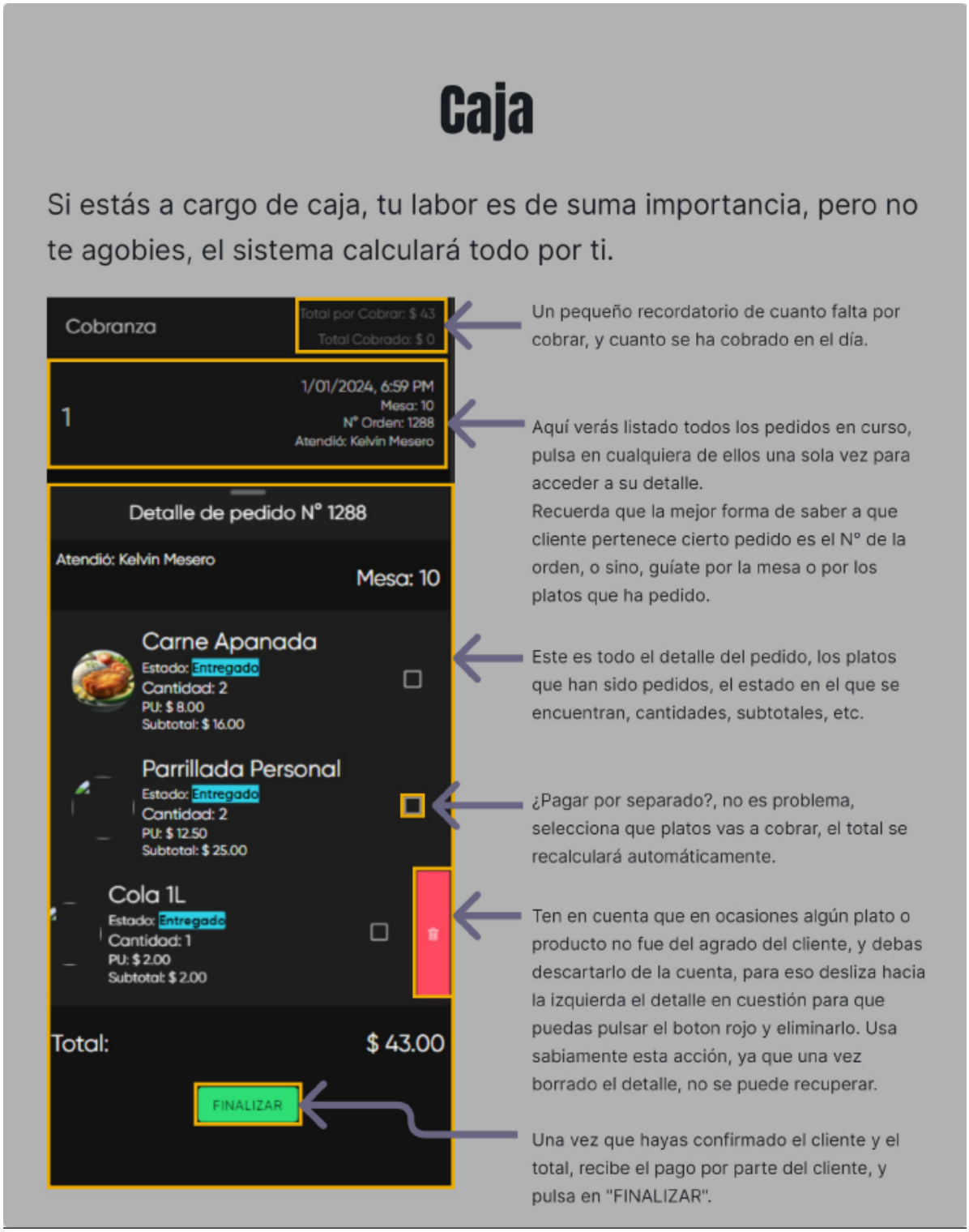


Figura A10. Módulo Caja

En la Figura A11 se muestra la introducción al módulo de Administración.

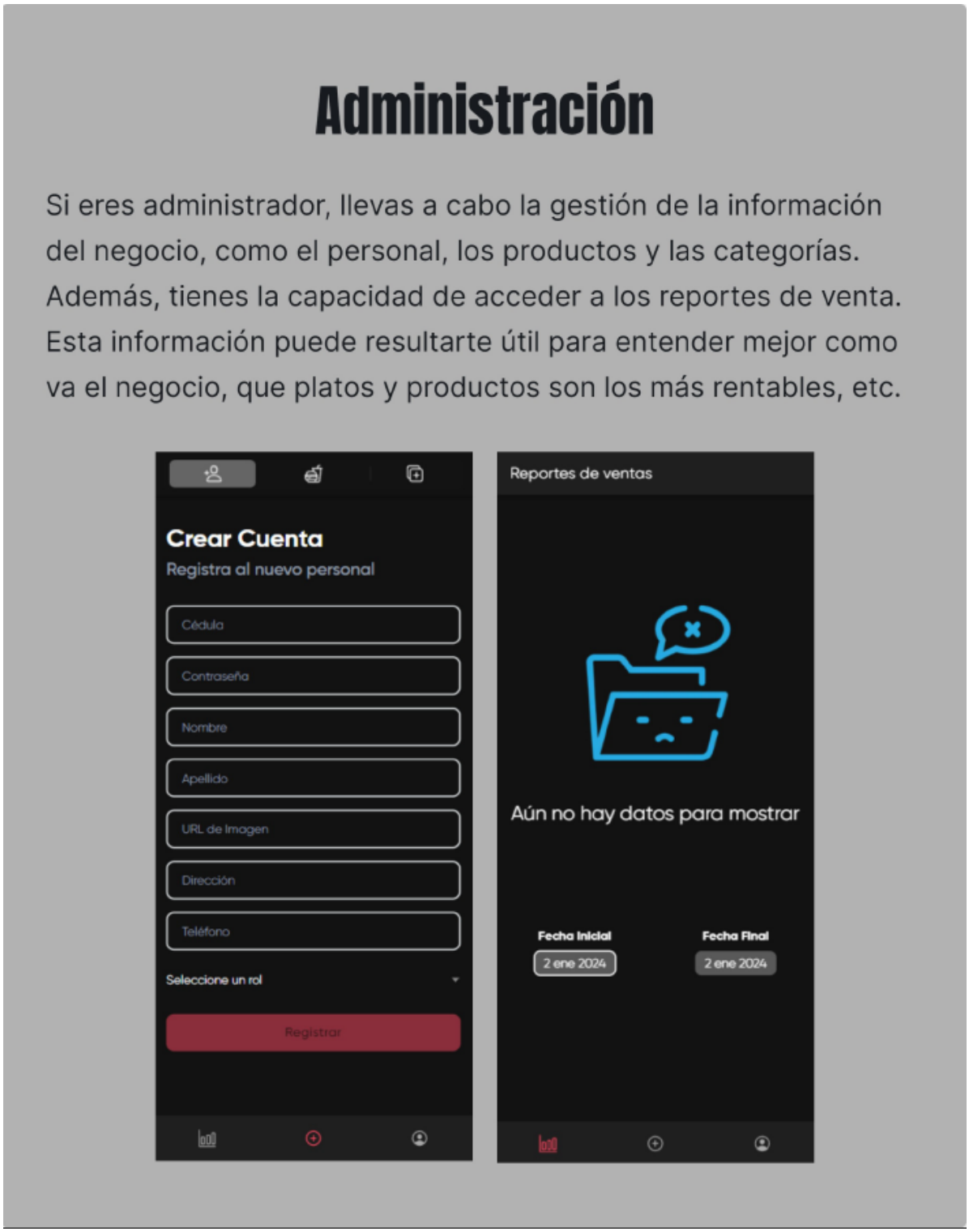


Figura A11. Módulo Administración

En la Figura A12 se muestra la introducción al módulo de Gestión.

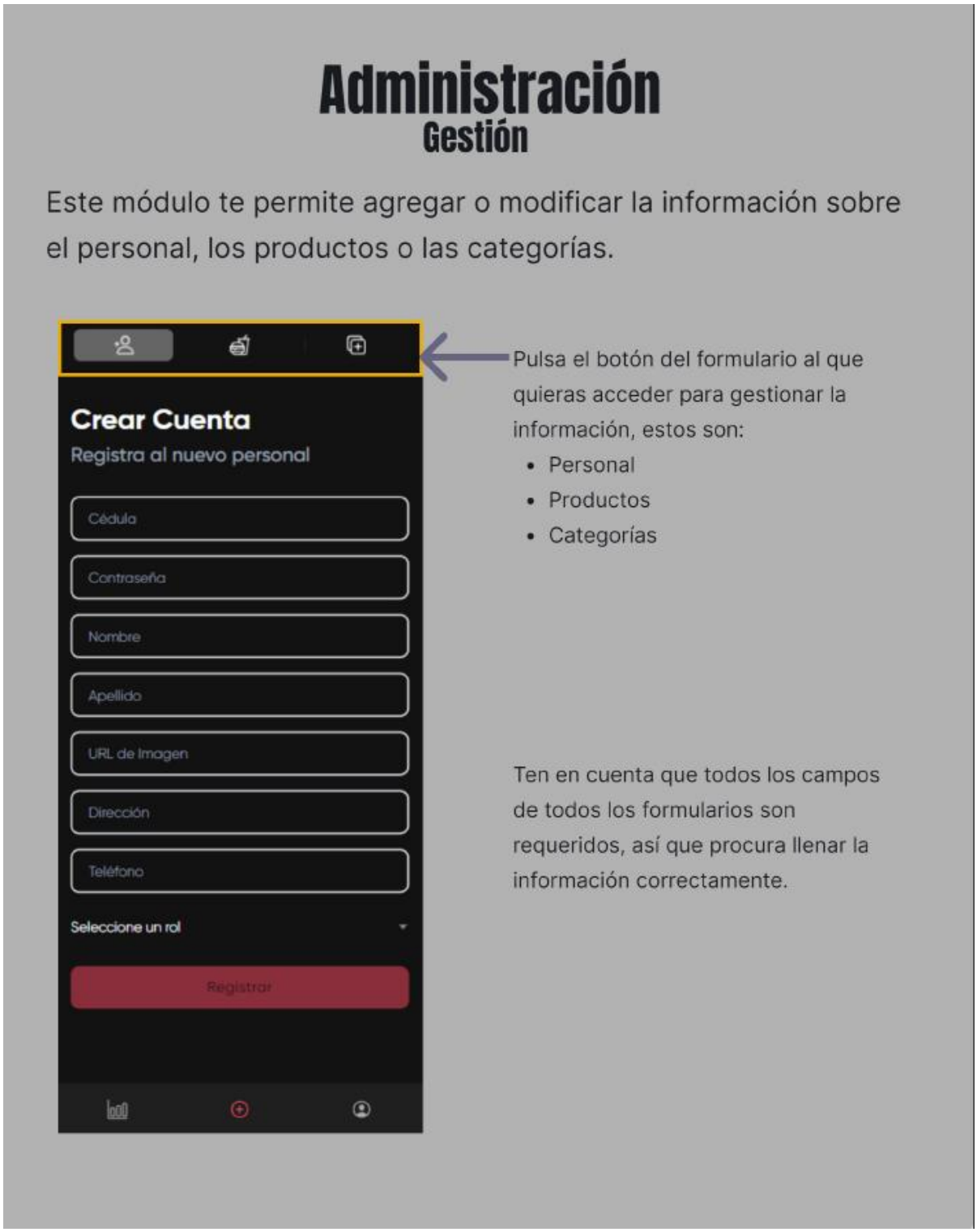


Figura A12. Módulo Gestión

En la Figura A13 se muestra el módulo de gestión del Personal.

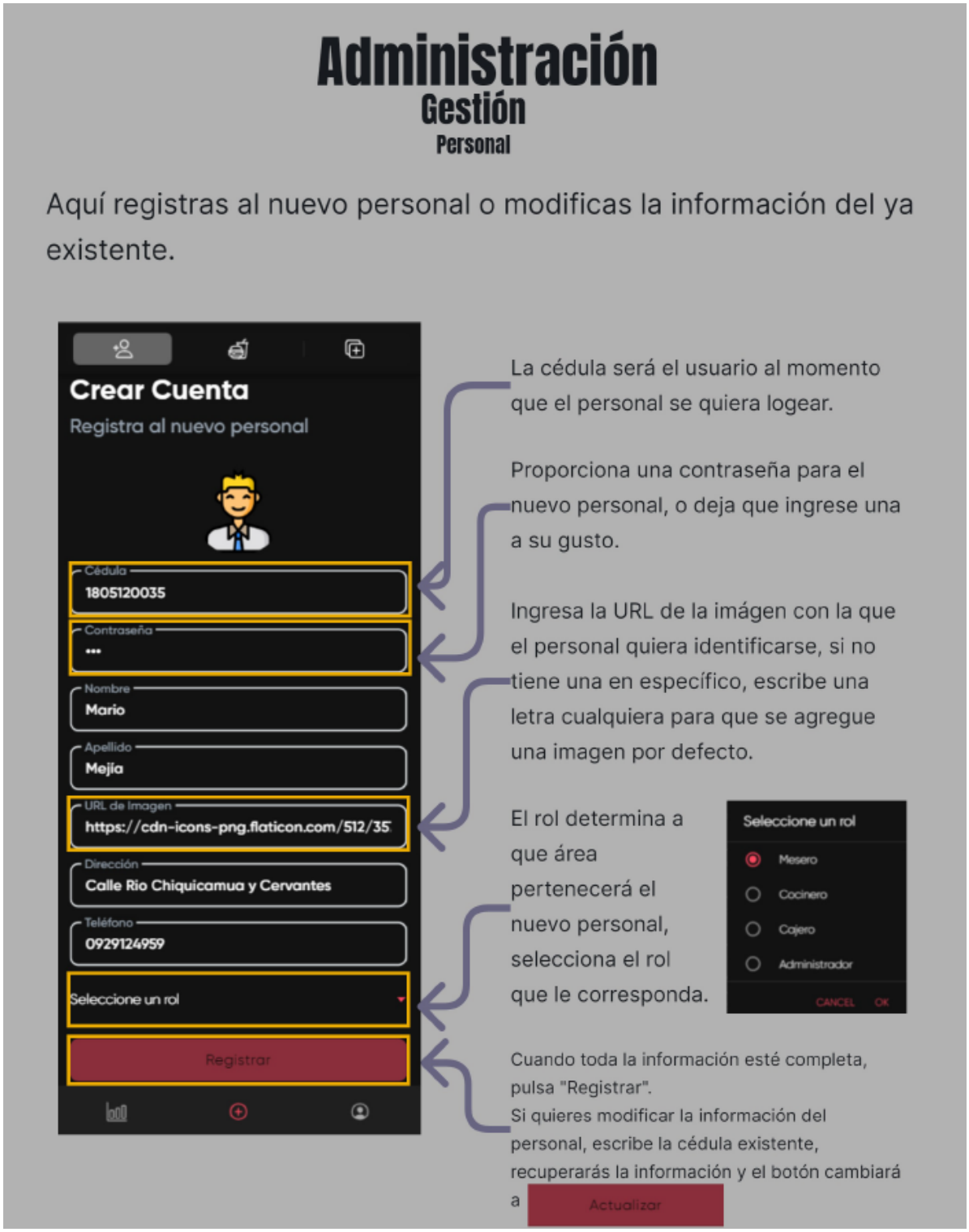


Figura A13. Módulo gestión del Personal

En la Figura A14 se muestra el módulo de gestión de los Productos.

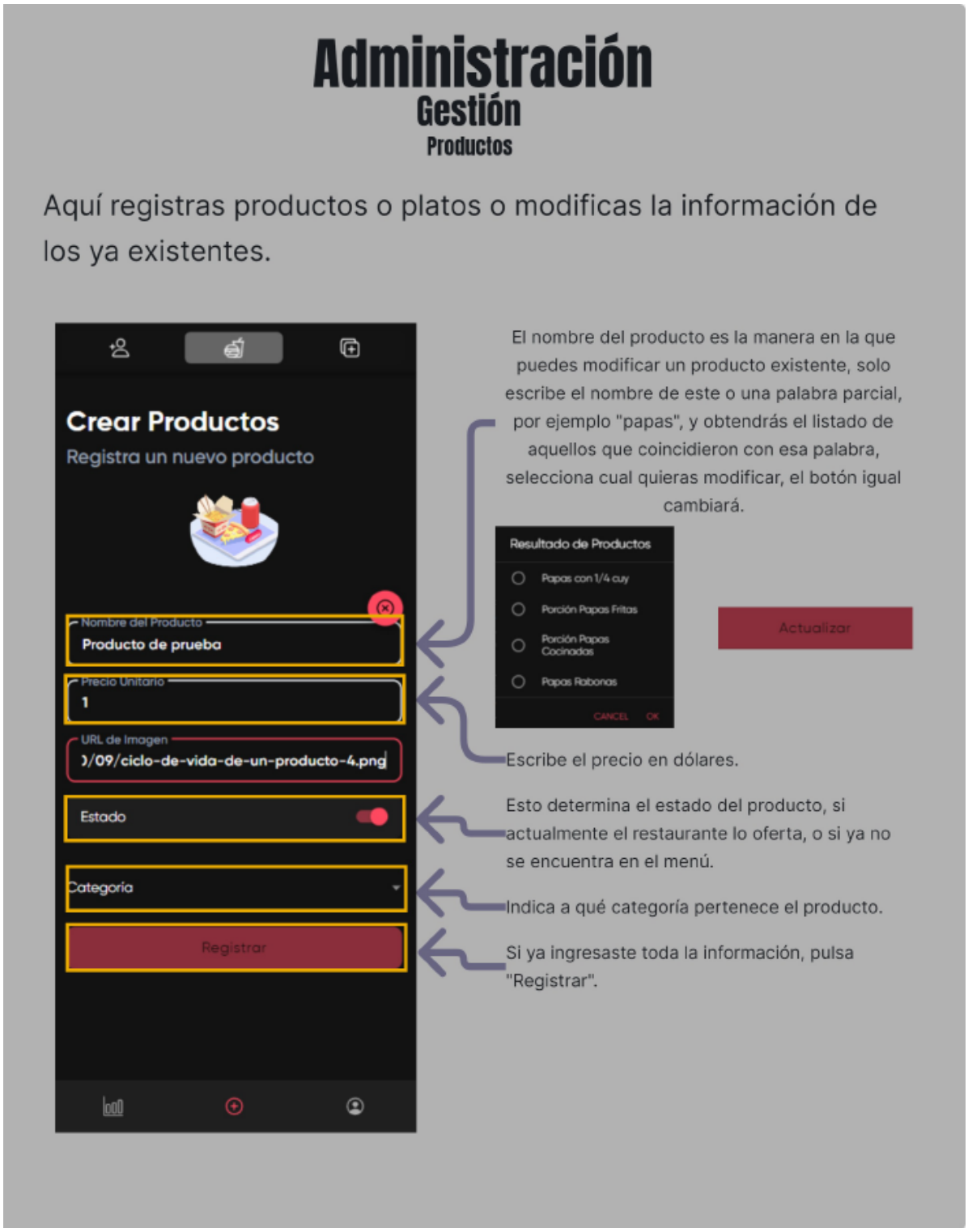


Figura A14. Módulo gestión de Productos

En la Figura A15 se muestra el módulo de gestión de Categorías.

Figura A15. Módulo gestión de Categorías

En la Figura A16 se muestra el módulo de Reportes.

Figura A16. Módulo Reportes

Anexo B. Repositorio GitHub App Móvil

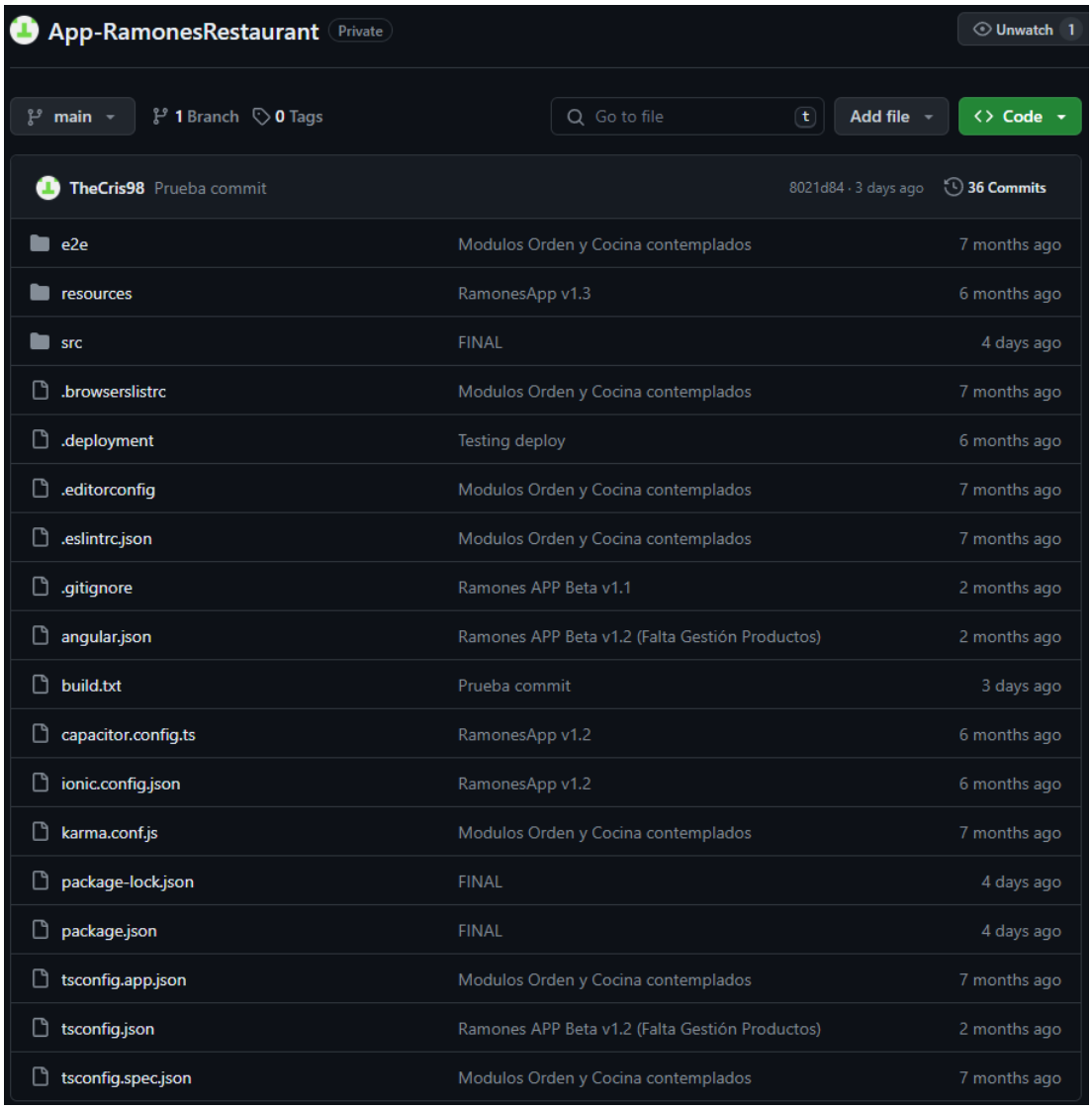


Figura B1. Archivos repositorio