



UNIVERSIDAD ESTATAL DE BOLÍVAR

**FACULTAD DE CIENCIAS ADMINISTRATIVAS, GESTIÓN
EMPRESARIAL E INFORMÁTICA**

CARRERA DE SOFTWARE

**TRABAJO DE INTEGRACIÓN CURRICULAR
PREVIO A LA OBTENCIÓN DEL TÍTULO DE
INGENIEROS EN SOFTWARE**

FORMA: PROYECTO TECNOLÓGICO

TEMA:

**APLICACIÓN MÓVIL PARA LA GESTIÓN EFICIENTE DE PEDIDOS EN
LOS LOCALES, DEL COMEDOR UNIVERSITARIO DE LA
UNIVERSIDAD ESTATAL DE BOLÍVAR, AÑO 2023.**

AUTORES:

**HENRY ALEXANDER OCAÑA PEÑA
LENON JOEL GANCHALA CRUZ**

DIRECTORA:

ING. MÓNICA BONILLA M.

GUARANDA – ECUADOR

2023

TEMA DEL PROYECTO TECNOLÓGICO

APLICACIÓN MÓVIL PARA LA GESTIÓN EFICIENTE DE PEDIDOS EN
LOS LOCALES DEL COMEDOR UNIVERSITARIO DE LA UNIVERSIDAD
ESTATAL DE BOLÍVAR, AÑO 2023.

DEDICATORIA

El presente trabajo de titulación va dedicado en primer lugar a Dios y a mi familia ya que con su apoyo incondicional he logrado cumplir una más de mis metas, además quiero dedicar a mis profesores y amigos quienes forjaron y guiaron en este camino de mi formación académica, de los cuales conservare en mis recuerdos consejos y enseñanzas que en mi persona lograron darme fortaleza para continuar y terminar mi carrera.

Henry Ocaña

El esfuerzo y dedicación que conlleva el presente trabajo lo dedico a mis padres y hermano que supieron apoyarme, brindándome sus consejos y vivencias guiaron mi camino para hacer realidad mi sueño de finalizar mi carrera, este logro no hubiese sido posible sin su infinito amor, apoyo incondicional y constante inspiración. Espero que por medio de estas palabras represente una pequeña muestra de gratitud.

Lenon Ganchala

AGRADECIMIENTO

Expresamos un agradecimiento a todos aquellos que contribuyeron en el desarrollo de este trabajo de titulación y entre ellos agradecemos a nuestra directora de tesis, la Ingeniera Mónica Bonilla, por su orientación profesional, paciencia e incansable dedicación. También expresamos un grato agradecimiento a nuestros pares académicos la Ing. Marisela Espina y Lic. Edgar Rivadeneira, por sus comentarios y sugerencias que enriquecieron enormemente nuestro trabajo de titulación.

Finalmente, agradecemos a la Universidad Estatal de Bolívar por brindarnos una base académica sólida las cuales las pusimos en práctica en el desarrollo del presente trabajo de titulación, y que serán de vital importancia en nuestras vidas profesionales.

Henry A. Ocaña; Lenon J. Ganchala

CERTIFICADO DE VALIDACIÓN

Ing. Mónica Bonilla, Ing. Maricela Espín y Lic. Edgar Rivadeneira, en su orden Director y Pares Académicos del Trabajo de Integración Curricular “APLICACIÓN MÓVIL PARA LA GESTIÓN EFICIENTE DE PEDIDOS EN LOS LOCALES DEL COMEDOR

UNIVERSITARIO DE LA UNIVERSIDAD ESTATAL DE BOLÍVAR, AÑO 2023.” desarrollado por los señores Lenon Joel Ganchala Cruz y Henry Alexander Ocaña Peña.

CERTIFICAN

Que, luego de revisado el Trabajo de Integración Curricular en su totalidad, cumple con las exigencias académicas de la carrera SOFTWARE, por lo tanto, autorizamos su presentación y defensa.

Guaranda, 5 de febrero del 2024



Ing. Mónica Bonilla
Director



Ing. Maricela Espín
Par Académico



Lic. Edgar Rivadeneira
Par Académico

a

DERECHOS DE AUTOR

Nosotros Ganchala Cruz Lenon Joel y Ocaña Peña Henry Alexander portadores de la Cédula de Identidad No 0250005261 y 0250075199 en calidad de autores y titulares de los derechos morales y patrimoniales del Trabajo de Titulación:
APLICACIÓN MÓVIL PARA LA GESTIÓN EFICIENTE DE PEDIDOS EN LOS LOCALES, DEL COMEDOR UNIVERSITARIO DE LA UNIVERSIDAD ESTATAL DE BOLÍVAR AÑO 2023 modalidad presencial, de conformidad con el Art. 114 del CÓDIGO ORGÁNICO DE LA ECONOMÍA SOCIAL DE LOS CONOCIMIENTOS, CREATIVIDAD E INNOVACIÓN, concedemos a favor de la Universidad Estatal de Bolívar, una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra, con fines estrictamente académicos. Conservamos a mi/nuestro favor todos los derechos de autor sobre la obra, establecidos en la normativa citada.

Así mismo, autorizamos a la Universidad Estatal de Bolívar, para que realice la digitalización y publicación de este trabajo de titulación en el Repositorio Digital, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

Los autores declaran que la obra objeto de la presente autorización es original en su forma de expresión y no infringe el derecho de autor de terceros, asumiendo la responsabilidad por cualquier reclamación que pudiera presentarse por esta causa y liberando a la Universidad de toda responsabilidad.



Lenon Joel Ganchala Cruz



Henry Alexander Ocaña Peña

ÍNDICE DE CONTENIDOS

TEMA DEL PROYECTO TECNOLÓGICO.....	1
DEDICATORIA.....	2
AGRADECIMIENTO.....	3
CERTIFICADO DE VALIDACIÓN.....	4
DERECHOS DE AUTOR (Acorde al modelo emitido por biblioteca general) .	4
INTRODUCCIÓN	11
RESUMEN.....	12
ABSTRACT.....	13
CAPÍTULO I.....	14
1.1. Tema.....	14
1.2. Descripción del Problema	14
1.3. Justificación.....	15
1.4. Objetivos:.....	16
Objetivo General	16
Objetivos Específicos	16
CAPÍTULO II	17
MARCO TEÓRICO.....	17
2.1. Antecedentes	17
2.2. Científico.....	19
2.3. Conceptual.....	20
2.4. Legal.....	22
2.5. Georeferencial	23
CAPÍTULO III.....	24
METODOLOGÍA	24
3.1. Metodología de Desarrollo de Software	24
CAPÍTULO IV.....	26

INGENIERÍA DEL PROYECTO	26
4.1. Análisis.....	26
4.1.1. Especificación de Requerimientos de Software.....	26
4.1.1.1. Introducción	26
4.1.1.2. Alcance del Producto.....	26
4.1.1.3. Definiciones, Acrónimos y Abreviaturas.....	26
4.1.1.4. Referencias.....	28
4.1.2. Descripción General del Producto	28
4.1.2.1. Perspectiva del producto	28
4.1.2.3. Características de los usuarios	30
4.1.2.4. Restricciones generales	30
4.1.3. Requerimientos de Usuario.....	31
4.2. Diseño	38
4.2.1. Arquitectura del Software	38
4.2.2. Diagramas	40
4.2.2.2. Diagrama de Secuencia.....	49
4.2.3. Diseño de la Interfaz	53
4.2.3.2. Interfaz de Pantallas	54
4.3. Programación	67
4.3.1. Definición del objetivo.....	67
4.3.2. Análisis del problema.....	67
4.3.3. Diseño del Algoritmo	67
4.3.4. Codificación	70
4.4. Pruebas	86
4.4.1. Análisis de requisitos.....	86
4.4.2. Planificación de pruebas	87
4.4.3. Diseño y desarrollo de casos de prueba	88

4.4.4. Configuración del entorno de prueba.....	89
Pruebas de Funcionalidad	94
Pruebas de Rendimiento.....	95
Pruebas de Seguridad	96
Pruebas de Disponibilidad.....	96
4.4.6. Prueba de cierre.....	97
CONCLUSIONES	98
RECOMENDACIONES	99
BIBLIOGRAFÍA.....	100
ANEXOS 1.....	102
Cronograma (Gantt)	102
Cronograma	1
ANEXOS 2.....	1
Presupuesto Ejecutado.....	1
ANEXOS 3.....	6
Instrumentos de recopilación de datos	6
ANEXOS 4.....	9
Introducción	13
Objetivo.....	13
Alcance.....	13
Restricciones	13
GUIA PASO A PASO EN LA APLICACIÓN MOVIL.....	14
1. ROL CLIENTE.....	14
1.1. Acceder a la aplicación	14
1.2. Registrar nuevo usuario	15
1.3. Sección de productos	15
1.4. Confirmar orden.....	16

1.5. Pago	19
2. Rol Restaurante	21
2.1. Iniciar sesión.....	21
2.2. Crear nueva categoría	22
2.3. Crear nuevo producto.....	23
2.4. Despachar Pedidos.....	24
3. Rol Repartidor	27
3.1. Iniciar sesión.....	27
3.2. Entregar los pedidos.....	27
ANEXOS 5.....	29
Manual Técnico.....	29
Introducción	32
Instalación y configuración del sistema	34
Controllers.....	35
.....	38
FrontEnd Estructura en Android Estudio	39
Clase Main	40
El pubspec.yaml	42
ANEXOS 6.....	43
Ejecución de las Pruebas.....	43
ANEXOS 7.....	47
Evidencias asistidas por la directora y los pares.	47

Índice De Tablas

Tabla 1	30
4.1.3.1. Requerimientos funcionales	31
4.1.3.2. Requerimientos no funcionales	35
Cliente	89
Propietario del local	91
Repartidor.....	93
Tabla de costos por módulos y componentes	4

Índice de figuras

Figura 1 Pantalla de Login	14
Figura 2 Pantalla de Registro	15
Figura 3 Pantalla de productos.....	16
Figura 4 Pasos para confirmar orden	17
Figura 5 Pasos para agregar la ubicación de entrega	18
Figura 6 Pasos para agregar el punto de referencia en el mapa de Google para crear mi dirección.....	19
Figura 7 Pasos para realizar el pago.....	20
Figura 8 Pantalla de mis pedidos en el estado pago.....	21
Figura 9 Inicia sesión y pasa a la pantalla de los roles.....	22
Figura 10 Pasos para crear una nueva categoría	23
Figura 11 Pasos para crear un nuevo producto	24
Figura 12 Pasos para despachar una orden	25
Figura 13 Pasos para agregar repartidor y despachar la orden.....	26
Figura 14 Pasos para iniciar sesión como repartidor	27
Figura 15 Pasos para realizar la entrega de la orden.....	28

INTRODUCCIÓN

Los locales del comedor de la Universidad Estatal de Bolívar, han optado por utilizar una aplicación móvil que les ayude a mejorar la gestión de pedidos y la difusión de sus menús, permitiendo despertar el interés de los clientes y optimizando los tiempos de atención, esta aplicación móvil les permite publicar el menú y actualizarlo en tiempo real dando la oportunidad a los clientes de realizar pedidos informados, ya que tendrán acceso a la imagen de la comida acompañado de una descripción el cual ayudara en la elección de la misma, además podrán elegir el método de pago disponible de cada local, logrando de esta manera mejorar la experiencia del cliente.

A lo largo de este documento se exponen los requerimientos obtenidos por los dueños de los locales del comedor, las herramientas tecnologías utilizadas, para el desarrollo de la aplicación móvil.

En el primer capítulo formulación general del proyecto se describe la problemática de los locales del comedor de la Universidad Estatal de Bolívar, al instante de ofrecer sus servicios de alimentación, la correspondiente justificación y los objetivos planteados que proporcionaron la dirección y enfoque para el cumplimiento del proyecto. El segundo capítulo se describe el marco teórico en el cual se revisaron antecedentes de desarrollo de aplicaciones móviles para servicios de alimentación, se revisaron conceptos de vital importancia y parámetros legales, además se detalla información georeferencial que fue primordial en la ejecución del proyecto. En el tercer capítulo de metodología, se analizaron las metodologías de desarrollo de software, de la cual se escogió la metodología ágil Programación Extrema (XP) debido a que se ajusta a las necesidades del desarrollo de la aplicación móvil “Comedor UEB”. Por último, en el cuarto capítulo se estableció la ingeniería del proyecto en el cual se aplicó la metodología, se realizó el plan de pruebas para garantizar la calidad de la aplicación móvil.

RESUMEN

Las aplicaciones móviles en el ámbito de los servicios de alimentación ofrecen algunas ventajas tanto al consumidor como al local, por ejemplo, permite realizar pedidos de manera rápida y conveniente, por otro lado, para los locales brinda una forma adicional de llegar a los consumidores y generar más ventas. Por ello el objetivo del presente trabajo de titulación es desarrollar e implementar una aplicación móvil que ayude a llevar una gestión eficiente de pedidos en los locales del comedor universitario de la Universidad Estatal de Bolívar, con la finalidad de que los locales puedan difundir el menú de sus productos, captando la atención de los estudiantes y profesores solucionando los problemas de una mala gestión de información sobre los servicios de alimentación que brindan. La aplicación móvil fue desarrollada con la metodología Programación Extrema (XP), además se utilizó una arquitectura por capas, para la codificación de la aplicación móvil se utilizó Android Studio y Flutter para la parte del Front-end y para la parte del Back-end Nodejs, con una base de datos creada en MySQL. La aplicación móvil cuenta con tres roles restaurante que puede crear categorías, productos y gestionar pedidos, repartidor encargado de entregar los pedidos y cliente que es quien realiza el pedido. En el documento se detalla el instrumento utilizado para la recolección de datos, las pruebas realizadas a la aplicación, así como el manual de usuario y manual técnico, con la finalidad de que la aplicación móvil Comedor UEB mantenga una gestión eficiente de pedidos en el comedor universitario con su uso.

Palabras clave: Servicios de alimentación, gestión eficiente de pedidos, arquitectura por capas.

ABSTRACT

Mobile applications in the field of food services offer some advantages to both the consumer and the premises, for example, it allows ordering to be placed quickly and conveniently, on the other hand, for the premises it provides an additional way to reach consumers and generate more sales. Therefore, the objective of this degree work is to develop and implement a mobile application that helps to carry out efficient order management in the university dining room premises of the State University of Bolívar, with the aim that the premises can disseminate the menu of their products, capturing the attention of students and teachers, solving the problems of poor information management about the food services they provide. The mobile application was developed with the Extreme Programming (XP) methodology, in addition a layered architecture was used, for the coding of the mobile application Android Studio and Flutter were used for the Front-end part and for the Back-end part Nodejs, with a database created in MySQL. The mobile application has three roles: restaurant that can create categories, products and manage orders, delivery person in charge of delivering the orders and customer who places the order. The document details the instrument used for data collection, the tests carried out on the application, as well as the user manual and technical manual, with the purpose of the UEB Dining Room mobile application maintaining efficient order management in the dining room. university with its use.

Keywords: Food services, efficient order management, layered architecture.

CAPÍTULO I

FORMULACIÓN GENERAL DEL PROYECTO

1.1. Tema

APLICACIÓN MÓVIL PARA LA GESTIÓN EFICIENTE DE PEDIDOS EN LOS LOCALES DEL COMEDOR UNIVERSITARIO DE LA UNIVERSIDAD ESTATAL DE BOLÍVAR, AÑO 2023.

1.2. Descripción del Problema

La nutrición en la educación universitaria tiene un impacto significativo, ya que afecta directamente el rendimiento académico, la concentración y el bienestar general de los estudiantes. El buen funcionamiento del cerebro se ve facilitado por una dieta saludable que proporcione los nutrientes necesarios, lo que mejora el aprendizaje y la retención de la memoria.

La Universidad Estatal de Bolívar cuenta con 267 docentes, 117 miembros del personal administrativo y 8000 estudiantes matriculados en el presente periodo académico mayo-septiembre 2023, quienes hacen uso de los servicios de alimentación que proveen los locales del comedor universitario, también existen en los alrededores de la institución varios negocios que ofertan este servicio.

Actualmente, los establecimientos que brindan servicios de alimentación en el comedor de la Universidad Estatal de Bolívar, no cuentan con una gestión eficiente de pedidos por lo que se generan aglomeraciones causando retrasos en la atención, en cada uno de los locales adscritos al comedor; además, no se mantiene actualizada la disponibilidad del menú de parte de los locales, ocasionando que el usuario pierda el interés de visitarlos.

Normalmente el menú que ofrece cada uno de los locales, son escritos en pizarras y promocionado en cuanto el cliente pasa por delante, esperando la posibilidad de dar mayor información al despertar el interés en el potencial cliente; en otras palabras, existe una mala gestión de la información, al no mostrarla con antelación y sobre todo de manera completa o integral; esto a su vez afecta de forma negativa

la experiencia del cliente, que posiblemente resultará en un alejamiento del servicio que se quiere promocionar.

1.3. Justificación

La falta de una gestión eficiente de pedidos en los locales del comedor de la Universidad Estatal de Bolívar ha generado aglomeraciones y retrasos en la atención a los clientes, lo que ha afectado negativamente tanto a los usuarios como al personal del comedor; además, la falta de información sobre la disponibilidad de los alimentos del menú ha llevado a una pérdida de interés por parte de los consumidores universitarios, quienes enfrentan la dificultad de visitar físicamente los locales para conocer el menú y consultar a los empleados para obtener más detalles. Estos problemas han impactado negativamente la experiencia del cliente y la eficiencia general del servicio.

Se propone implementar una aplicación móvil para la gestión de pedidos en el comedor universitario, esta solución permitirá mejorar el proceso de pedidos al reducir las aglomeraciones y optimizar los tiempos de atención. Los usuarios podrán acceder al menú completo, conocer la disponibilidad de alimentos en tiempo real y realizar pedidos con anticipación. Además, la aplicación proporcionará información adicional, como ingredientes, alérgenos y opciones especiales para ayudar a los usuarios a tomar decisiones informadas.

Con el tiempo, la tecnología de aplicaciones móviles ha experimentado un gran crecimiento tanto a nivel nacional como internacional. Sin embargo, su auge más significativo se produjo durante la crisis sanitaria mundial, según señaló la Organización Mundial de la Salud (OMS) en un informe realizado por Danyluk, Seymour, Yavelak y Christian en el año 2020.

La presente investigación aportará en la línea de Ingeniería De Software, Redes y Telecomunicaciones además aporta a las sub líneas de investigación como el Diseño e implementación de sistemas de información.

1.4. Objetivos:

Objetivo General

Implementar una aplicación móvil para la gestión de pedidos en el comedor universitario de la Universidad Estatal de Bolívar, año 2023.

Objetivos Específicos

- Realizar un análisis de requerimientos funcionales y no funcionales para identificar las necesidades de la aplicación móvil.
- Diseñar la arquitectura de la aplicación y definir la interfaz de usuario.
- Realizar pruebas de calidad para garantizar el correcto funcionamiento de la aplicación.

CAPÍTULO II

MARCO TEÓRICO

2.1. Antecedentes

La tecnología ha ido evolucionando a través del tiempo de forma increíble haciendo nuestra vida mucho más cómoda. En la actualidad las aplicaciones móviles están ganando territorio con fuerza a nivel mundial, tanto así que ya son parte de nuestra vida diaria, demostrando ser de gran utilidad para dueños de locales y empresarios, permitiéndoles administrar sus negocios o actividades personales de manera ágil y eficiente.

Flores (2021) en su trabajo hace alusión al desarrollo e implementación de una aplicación móvil de la clase Mercado Público B2B que permitirá a los productores mejorar, ampliar, organizar y manejar relaciones comerciales con los clientes, entre ellos resaltan las empresas gastronómicas, además permite optimizar la gestión de pedidos. Se puso en marcha un servicio API REST con CodeIgniter para facilitar operaciones CRUD en la información de la base de datos, además utilizaron Flutter SDK para el desarrollo de la aplicación móvil. Finalmente realizaron las pruebas necesarias y compararon el antes y el después de poner en funcionamiento la aplicación.

Según Mocha y Ochoa (2022) en su tesis cuyo objetivo principal fue desarrollar una aplicación móvil y una aplicación web para poder automatizar y dar un eficiente servicio al cliente, con la finalidad de prevenir las aglomeraciones dentro de los restaurantes en Cuenca. Esta aplicación posee funciones específicas como la automatización de pedidos tanto en dispositivos móviles como para la web, verifica el estado del pedido indicando si está en espera, en proceso o finalizado, incluye un listado de productos, pedidos y facturas controlado con el IVA del 12%, un carrito de compras en tiempo real, notificaciones del pedido entre el personal del restaurante y el cliente. Todo lo descrito se ve plasmado en dos aplicaciones con distintas funciones, la una genera un dashboard para el productor y la otra aplicación proporciona una visualización de los pedidos para los clientes del restaurante.

Villanueva (2022) en su trabajo aborda el proceso de desarrollo de una aplicación móvil para la empresa Transporte Sanmartí SA, con el objetivo de permitirle a la

empresa gestionar los pedidos en tiempo real asignados a cada transportista. El producto final permitirá un riguroso seguimiento del estado de los pedidos y un intercambio recíproco de información entre el empleado y el departamento, manteniendo una comunicación constante. Además, brindará distintas funcionalidades a los transportistas, así como ingreso de datos, descarga de documentos, mensajería en tiempo real, incluido la ubicación de puntos de interés tales como el punto de carga o de descarga, entre otros.

Según Allain (2020), en su investigación aborda el diseño e implementación de una aplicación informática para la gestión eficiente de comandas en el restaurante Gaitano Resto Bar de Ica, el cual enfrenta una serie de problemas relacionados con la gestión de comandas. Por lo cual su objetivo primordial es mejorar esta gestión proporcionando una solución informática eficiente que contribuya a optimizar los procesos internos del establecimiento y brinde una experiencia satisfactoria tanto para el personal del restaurante como para los clientes.

2.2. Científico

En el campo del desarrollo de software, las metodologías ágiles han surgido como una solución eficiente a los desafíos de la gestión de proyectos en un entorno de constante cambio y avance tecnológico. Estos métodos se basan en principios colaborativos y adaptativos y tienen como objetivo la creación de valor iterativo e incremental.

La agilidad del desarrollo de software se centra en la flexibilidad, la comunicación eficaz y la respuesta rápida al cambio, lo que permite a los equipos de desarrollo adaptar su enfoque a medida que cambian las necesidades del proyecto y del cliente. Esto contrasta con los enfoques tradicionales, más rígidos y lineales, como el método de cascada, que puede dificultar la adaptación al cambio y retrasar la entrega del producto.

La metodología ágil tiene una amplia gama de enfoques, cada uno con sus propias características y prácticas. Uno de esos enfoques es la metodología de Programación Extrema (XP), cuyos principales objetivos son la calidad del software y la satisfacción del cliente.

Según Pressman un libro clásico de la ingeniería del software, establece que “en la programación extrema, los requerimientos se expresan como escenarios (llamados historias de usuario), que se implementan directamente como una serie de tareas. Los programadores trabajan en pares y antes de escribir el código desarrollan pruebas para cada tarea. Todas las pruebas deben ejecutarse con éxito una vez que el nuevo código se integre en el sistema.” (Pressman, 2011, p.65)

El desarrollo incremental se apoya en pequeñas y frecuentes liberaciones del sistema. Los requerimientos se fundamentan en simples historias del cliente, o bien, en escenarios usados como base para decidir qué funcionalidad debe incluirse en un incremento del sistema. (Pressman, 2011, p.65)

La inclusión del cliente se apoya a través de un enlace continuo con el cliente en el equipo de desarrollo. El representante del cliente participa en el desarrollo y es responsable de definir las pruebas de aceptación para el sistema (Pressman, 2011, p.65).

Las personas, no los procesos, se basan en la programación en pares, en la propiedad colectiva del código del sistema y en un proceso de desarrollo sustentable que no incluya jornadas de trabajo excesivamente largas. (Pressman, 2011, p.65)

El cambio se acepta mediante liberaciones regulares del sistema a los clientes, desarrollo de primera prueba, refactorización para evitar degeneración del código e integración continua de nueva funcionalidad. (Pressman, 2011, p.65)

Mantener la simplicidad se logra mediante la refactorización constante, que mejora la calidad del código, y con el uso de diseños simples que no anticipan innecesariamente futuros cambios al sistema. (Pressman, 2011, p.65)

Según Rodríguez (2011) Mobile-D es una metodología diseñada para agilizar el proceso de desarrollo en equipos pequeños, centrándose específicamente en la creación de aplicaciones móviles, consta de varias etapas, incluyendo exploración, inicialización, fase de producto, fase de estabilización y fase de pruebas, además asigna días específicos para la planificación y entrega en cada etapa.

Durante la etapa de exploración, se pone énfasis en la planificación y se establecen los conceptos fundamentales del proyecto, definiendo el alcance y las funcionalidades deseadas.

La etapa de inicialización implica la configuración del proyecto, identificando y preparando los recursos necesarios. Aquí, se dedica un día a la planificación, mientras que el resto del tiempo se dedica al trabajo y a la publicación.

La fase de producto se divide en subetapas iterativas que utilizan el desarrollo dirigido por pruebas (TDD). Antes de implementar una funcionalidad, se crea una prueba para verificar su funcionamiento, centrándose en la implementación.

En la etapa de estabilización, se llevan a cabo acciones de integración para unificar posibles módulos separados en una única aplicación.

Por último, en la fase de pruebas, se suspende completamente el desarrollo y se realiza una serie de pruebas para obtener una versión estable, de acuerdo con los requisitos acordados con el cliente en las etapas iniciales, de ser necesario, se corrigen errores, pero no se añade ninguna funcionalidad adicional.

2.3. Conceptual

API: “Una API es el acrónimo de Interfaz de Programación de Aplicaciones, que es un intermediario de software que permite que dos aplicaciones se comuniquen entre sí. Cada vez que usas una aplicación como Facebook, envías un mensaje instantáneo o consultas el clima en tu teléfono, estás usando una API” (Mulesoft, 2021).

Aplicación informática: “Una aplicación informática es un programa diseñado para realizar una serie de tareas específicas en un entorno informático” (TechTerms, 2021).

Autenticación y autorización: “La autenticación es el proceso de verificar la identidad de un usuario, mientras que la autorización es el proceso de otorgar o denegar acceso a recursos específicos basándose en la identidad del usuario” (Microsoft, 2021).

Base de datos: “Una base de datos es un conjunto organizado de datos almacenados en una computadora o en un servidor, diseñado para permitir la búsqueda y recuperación eficiente de información” (Oracle, 2021).

Experiencia del usuario (UX): “La experiencia del usuario se refiere a cómo una persona se siente al interactuar con un sistema, ya sea una aplicación web, un sitio web, un software de escritorio o cualquier otro producto o servicio” (Nielsen Norman Group, 2021).

Geolocalización: “La geolocalización es la capacidad de determinar la ubicación geográfica de un dispositivo conectado a Internet” (Techopedia, 2021).

Gestión de inventario y menús: “La gestión de inventario y menús implica el seguimiento y control de los productos y opciones disponibles en un restaurante, con el fin de garantizar que los clientes tengan acceso a información precisa y actualizada” (Toast, 2021).

Interfaz de usuario (UI): “La interfaz de usuario es el medio por el cual un usuario interactúa con una aplicación o un sitio web, utilizando elementos como botones, menús y formularios para navegar y realizar acciones” (Usability.gov, 2021).

Modelo Vista Controlador (MVC): “El patrón Modelo-Vista-Controlador (MVC) es un enfoque de desarrollo que separa la lógica de la aplicación en tres

componentes interconectados: el modelo, la vista y el controlador” (Microsoft, 2021).

Notificaciones push: “Las notificaciones push son mensajes cortos que se envían desde una aplicación a los dispositivos móviles de los usuarios para informarles sobre novedades, promociones u otras actualizaciones relevantes” (OneSignal, 2021).

Pasarelas de pagos: “Una pasarela de pago es un servicio que permite a los comerciantes aceptar pagos en línea mediante tarjetas de crédito, débito u otros métodos electrónicos” (PayPal, 2021).

Sistema de pedidos: “Un sistema de pedidos es una aplicación o plataforma que permite a los clientes seleccionar productos o servicios, agregarlos al carrito y completar el proceso de pago” (Shopify, 2021).

Sistema de seguimiento de pedidos: “Un sistema de seguimiento de pedidos permite a los clientes rastrear el estado y progreso de sus pedidos en tiempo real, desde la confirmación hasta la entrega” (FedEx, 2021).

2.4. Legal

El desarrollo de aplicaciones móviles para pedidos de alimentos generalmente se rige por las mismas leyes y reglamentos que se aplican al comercio electrónico y los servicios digitales en Ecuador estas incluyen la Ley de Comercio Electrónico, la Ley de Protección al Consumidor y la Ley de Protección de Datos Personales, entre otras.

La norma ISO 27001: Ayuda a las organizaciones a gestionar la seguridad de los activos sensibles, como datos financieros, propiedad intelectual, registros de empleados, datos de clientes y otra información sensible. Obliga a las empresas a aplicar la seguridad de la información, por lo que se reducen los posibles riesgos de robos y filtraciones de datos.

Ley Orgánica de Comunicación (LOC): Esta ley ecuatoriana regula los medios de comunicación, incluidas las aplicaciones móviles que brindan servicios de comunicación. Si una aplicación móvil ofrece servicios de medios de comunicación en Ecuador, debe cumplir con los requisitos establecidos en la LOC, como la protección de datos personales y la responsabilidad en la difusión de información.

La Ley Orgánica de Datos Personales (LODP): Es la ley que regula la protección de los datos personales en Ecuador. Fue aprobada el 27 de enero de 2021 y publicada en el Registro Oficial No. 1232 el 10 de febrero de 2021.

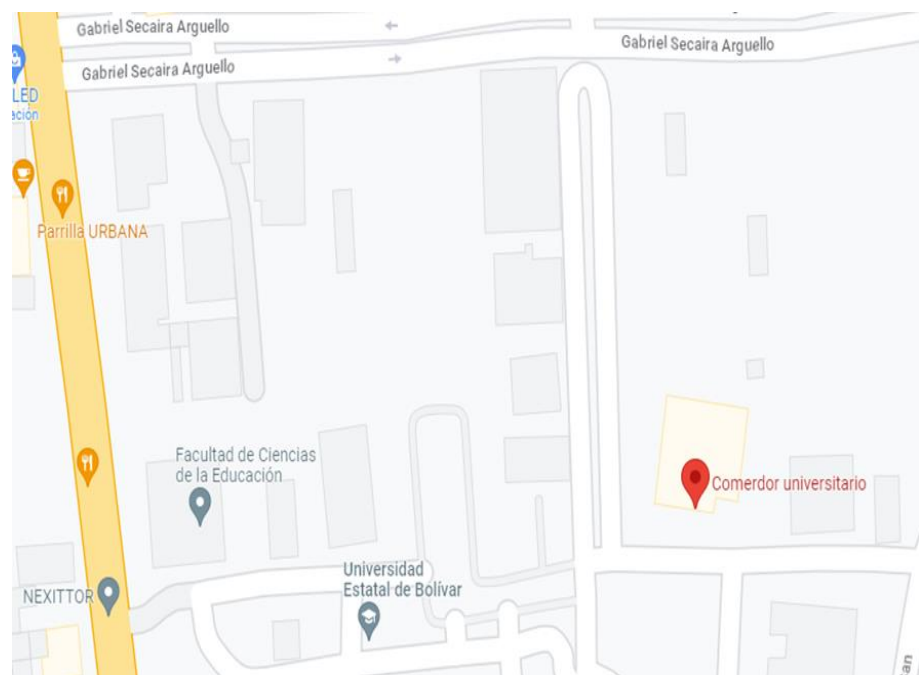
ACM/IEEE-CS: Fuerza de trabajo conjunta acerca de ética y prácticas profesionales de la ingeniería de software

2.5. Georeferencial

Imagen N° 1

Mapa de la Universidad Estatal de Bolívar

Comedor Universitario



Google. (s.f.). [Comedor UEB]. Recuperado el 21 de septiembre de 2023 de

<https://maps.app.goo.gl/QAzDm9UHAsAV12sF7>

Opciones de servicio: Consumo en el lugar

La Universidad Estatal de Bolívar tiene su campus matriz en la dirección Av. Ernesto Che Guevara s/n y Av. Gabriel Secaira en Guaranda, Ecuador, Latitud-1,5707° o 1° 34' 15" sur, Longitud-79,00627° o 79° 0' 23" oeste.

CAPÍTULO III

METODOLOGÍA

3.1. Metodología de Desarrollo de Software

Para el desarrollo del presente proyecto tecnológico se aplicó la metodología XP (Programación Extrema, por sus siglas en inglés), ya que es un enfoque ágil de desarrollo de software y se centra en la aportación rigurosa de cada integrante del equipo, se adapta a los cambios y se realiza una entrega continua de software funcional, además de permitir una participación activa del cliente durante todos los procesos de desarrollo.

Esta metodología consta de las siguientes fases:

Planificación: En esta fase se realizó un análisis de los recursos y el tiempo requeridos para llevar a cabo el proyecto. Se definieron los requisitos en colaboración con el personal del comedor universitario, comprendiendo sus necesidades y expectativas. Se priorizaron las historias de usuarios considerando criterios como el valor para el negocio, complejidad técnica y dependencias entre funcionalidades.

Diseño e implementación iterativa: Durante esta etapa, se diseñó la arquitectura de la aplicación móvil, tomando decisiones técnicas clave, como la selección de tecnologías adecuadas y la definición de la estructura general del sistema. Se implementó de manera iterativa, añadiendo nuevas funcionalidades y mejorando continuamente el software.

Pruebas y retroalimentación: Se puso énfasis en llevar a cabo pruebas exhaustivas de todas las funcionalidades de la aplicación. Se diseñaron casos de prueba relevantes y se abordaron diferentes escenarios para garantizar la calidad del software. Además, se recopiló y analizó la retroalimentación de los usuarios de manera efectiva para identificar áreas de mejora y realizar ajustes pertinentes.

Lanzamiento y puesta en producción: En esta fase, se enfocó en preparar el entorno de producción y configurar los servidores necesarios para el lanzamiento

de la aplicación. Se consideraron aspectos de seguridad, escalabilidad y rendimiento al implementar la aplicación en un entorno real.

3.2. Técnicas e Instrumentos de Recopilación de Datos

Para definir los requisitos de la aplicación móvil se utilizó como técnica la entrevista para recabar información detallada y específica de los propietarios de los locales del comedor universitario de la Universidad Estatal de Bolívar.

CAPÍTULO IV

INGENIERÍA DEL PROYECTO

4.1. Análisis

4.1.1. Especificación de Requerimientos de Software

4.1.1.1. Introducción

4.1.1.2. Alcance del Producto

La aplicación móvil Comedor UEB permite mejorar la eficiencia en la gestión de pedidos en los locales del comedor universitario, con el objetivo de mejorar la atención del cliente. Entre sus funcionalidades y características se incluyen el registro de usuario, despliegue del menú y actualizado en tiempo real, información detallada sobre la disponibilidad y descripción de los alimentos, así como la posibilidad de realizar y personalizar pedidos. Además, ofrece opciones de pago seguras, notificaciones push sobre el estado del pedido y geolocalización para facilitar la entrega.

4.1.1.3. Definiciones, Acrónimos y Abreviaturas

- **NodeJS:** Plataforma de desarrollo de aplicaciones en tiempo de ejecución de JavaScript del lado del servidor.
- **Backend:** La parte de una aplicación o sistema que se encarga de la lógica y procesamiento detrás de escena, generalmente en el servidor.
- **Tablas en la base de datos:** Estructuras organizadas en filas y columnas utilizadas para almacenar y organizar datos en un sistema de gestión de bases de datos relacional.
- **SQL:** Structured Query Language (Lenguaje de Consulta Estructurado), es un lenguaje de programación utilizado para administrar y consultar bases de datos relacionales.

- **Socket.IO:** Biblioteca de JavaScript que permite la comunicación bidireccional en tiempo real entre el cliente y el servidor a través de websockets.
- **JSON Web Tokens (JWT):** Estándar abierto para crear tokens de acceso seguro y compactos que se utilizan para autenticar y autorizar a los usuarios en aplicaciones web y APIs.
- **Mapas de Google:** Servicio de mapas en línea proporcionado por Google, que permite mostrar mapas interactivos, obtener direcciones y realizar búsquedas geográficas.
- **Notificaciones Push:** Mensajes enviados a dispositivos móviles o computadoras, que pueden ser recibidos incluso cuando la aplicación no está activa, para informar sobre eventos o actualizaciones relevantes.
- **Rutas en el mapa:** Trayectoria o camino representado en un mapa, que puede ser trazado desde un punto de origen (A) hasta un punto de destino (B).
- **Marcadores personalizados:** Símbolos o iconos personalizados colocados en un mapa para señalar ubicaciones específicas o destacar puntos de interés.
- **Firestore Storage:** Servicio en la nube de almacenamiento de archivos proporcionado por Firebase, que permite guardar y recuperar archivos, como imágenes, de manera segura.
- **Relación de datos:** Vínculo o conexión entre entidades o tablas en una base de datos relacional, a menudo basado en claves primarias y claves foráneas.
- **Navegación con GPS:** Utilización de tecnología GPS (Sistema de Posicionamiento Global) para determinar la ubicación actual de un dispositivo y proporcionar indicaciones de navegación para llegar a un destino específico.
- **Geolocalización:** Proceso de determinar la ubicación geográfica precisa de un dispositivo o usuario utilizando técnicas como el GPS, redes celulares o direcciones IP.

- **Peticiones HTTP:** Solicitudes realizadas por un cliente a un servidor utilizando el protocolo de transferencia de hipertexto (HTTP), para obtener o enviar datos.
- **Android Studio:** Entorno de desarrollo integrado (IDE) utilizado para desarrollar aplicaciones Android.
- **Visual Studio Code:** Editor de código fuente desarrollado por Microsoft, utilizado para escribir y depurar código en varios lenguajes de programación.
- **Postman:** Herramienta que permite enviar peticiones HTTP a servicios web y API para probar, depurar y documentar las interacciones entre cliente y servidor.

4.1.1.4. Referencias

IEEE recommended practice for software requirements specifications IEEE Std 830,1993

4.1.2. Descripción General del Producto

4.1.2.1. Perspectiva del producto

Comedor UEB es una aplicación móvil de venta de comida que permite publicar el menú que ofrecen los locales del comedor de la Universidad Estatal de Bolívar con información detallada de los productos, además permite a los clientes realizar pedidos y seleccionar el método de pago tales como transferencia y pago en efectivo.

4.1.2.2. Funciones del producto

La aplicación móvil cuenta con múltiples funcionalidades diseñadas para mejorar la experiencia del cliente, además de facilitar la compra de comida en los locales del comedor de la Universidad Estatal de Bolívar, algunas de las funcionalidades más destacadas son:

Registro de Usuario: permite ingresar la información del propietario de cada local del comedor universitario, y de los clientes

para crear la sesión en la aplicación y soliciten servicios de alimentación, por lo cual una vez registrado tendrá acceso a las funcionalidades de la aplicación.

Publicación de Menú: permite a los locales actualizar su oferta en tiempo real subiendo información de los productos tales como imagen, cantidad, descripción y precio, ayudando a los clientes a seleccionar la mejor opción del producto de acuerdo a sus preferencias.

Gestión de Pedidos: permite solicitar el servicio de alimentación al cliente considerando el menú de cada local y además cuenta con envío de pedido y consumo en el local.

Métodos de Pagos: le permite al cliente escoger el método de pago más conveniente: transferencia y pago en efectivo.

Notificaciones Push: permite al propietario del local enviar notificaciones en tiempo real sobre el estado de los pedidos.

4.1.2.3. Características de los usuarios

Tabla 1

Características de los Usuarios

TIPO DE USUARIO	DESCRIPCIÓN	IMPORTANCIA (1,2,3,4,5)
ADMINISTRADOR	Encargado de validar la información de los usuarios de la aplicación	5
PROPIETARIO DEL LOCAL	Encargado de subir información de los productos en el menú, validación del pago, emitir notificaciones del pedido y despacho del producto	5
CLIENTE	Debe registrarse en el sistema, puede observar el menú de los locales y realizar pedidos.	5
DELIVERY	Encargado de entregar los pedidos al cliente que solicita el servicio	5

Ganchala L, Ocaña H. *Características de los usuarios*; 2023

4.1.2.4. Restricciones generales

- La aplicación móvil está destinada para la Matriz de la Universidad Estatal de Bolívar.
- Su uso será con conexión a internet.
- La aplicación está disponible para Android.

4.1.3. Requerimientos de Usuario

4.1.3.1. Requerimientos funcionales

ID RF-01	Registro de Usuario (Propietario)
Actores	Dueño del local
Interfaz	Login
Precondición	El usuario debe ingresar sus datos personales en la aplicación, por ejemplo, cédula, nombre, apellido, email, teléfono, contraseña, imagen del local
Descripción	El usuario debe poder registrarse o iniciar sesión en la aplicación utilizando su correo electrónico y contraseña
Postcondición	El usuario ha iniciado sesión y puede acceder a las funcionalidades de la aplicación

Ganchala L, Ocaña H. *Requerimiento registro de usuario*; 2023

ID RF-02	Registro de Usuario (Cliente)
Actores	Cliente
Interfaz	Login
Precondición	El usuario debe ingresar sus datos personales en la aplicación, por ejemplo, cédula, nombre, apellido, email, teléfono, contraseña.
Descripción	El usuario debe poder registrarse o iniciar sesión en la aplicación utilizando su correo electrónico y contraseña
Postcondición	El usuario ha iniciado sesión y puede acceder a las funcionalidades de la aplicación

Ganchala L, Ocaña H. *Requerimiento registro de cliente*; 2023

ID RF-03	Descripción del producto
Actores	Dueño del local
Interfaz	Formulario de edición de productos
Precondición	El dueño del local debe tener una cuenta y haber iniciado sesión en la aplicación.
Descripción	La aplicación debe permitir al dueño del local agregar o editar la descripción de los productos, la cual debe incluir la información detallada sobre el producto, precio, cantidad y una imagen de la comida.
Postcondición	La descripción del producto se guarda en el sistema y está disponible para que los clientes la vean.

Ganchala L, Ocaña H. *Requerimiento Descripción del producto; 2023*

ID RF-04	Publicación del Menú
Actores	Dueño del local
Interfaz	Edición de menú
Precondición	El dueño del local debe tener una cuenta y haber iniciado sesión en la aplicación.
Descripción	La aplicación debe permitir al dueño del local agregar y actualizar los platos y bebidas disponibles en el menú del día.
Postcondición	El menú se actualiza en la aplicación y está disponible para que los clientes lo vean.

Ganchala L, Ocaña H. *Requerimiento Publicación del menú ; 2023*

ID RF-05	Notificaciones Push
Actores	Dueño del local
Interfaz	Interfaz del dueño del local de la aplicación móvil
Precondición	El dueño del local debe tener una cuenta y haber iniciado sesión en la aplicación.
Descripción	La aplicación debe permitir al dueño del local enviar notificaciones a los clientes sobre el estado de su pedido y el estado de los productos, (en stock o agotado)
Postcondición	El cliente recibe una notificación sobre el estado de su pedido y el stock de los productos.

Ganchala L, Ocaña H. *Requerimiento Notificaciones push; 2023*

ID RF-06	Método de Pago
Actores	Cliente
Interfaz	Interfaz de cliente de la aplicación móvil
Precondición	El usuario ha agregado platos al carrito de compras y desea realizar el pedido
Descripción	El usuario debe poder realizar el pago de forma segura utilizando métodos de pago admitidos, como transferencia bancaria o pago en efectivo
Postcondición	El pago se procesa correctamente y se confirma el pedido

Ganchala L, Ocaña H. *Requerimiento Métodos de pago; 2023*

ID RF-07	Realizar pedido
Actores	Cliente
Interfaz	Interfaz de cliente de la aplicación móvil
Precondición	El cliente debe haber seleccionado los platos y bebidas que desea ordenar.
Descripción	La aplicación debe permitir a los clientes realizar un pedido seleccionando los platos y bebidas que desean ordenar.

Postcondición	El pedido se procesa en la aplicación para ser preparado en el local.
----------------------	---

Ganchala L, Ocaña H. *Requerimiento Realizar pedidos; 2023*

ID RF-08	Soporte al cliente
Actores	Cliente
Interfaz	Interfaz del cliente de la aplicación móvil
Precondición	El usuario tiene una pregunta, consulta o problema relacionado con la aplicación o el servicio
Descripción	El usuario debe poder ponerse en contacto con el servicio al cliente a través de un número de teléfono o un formulario de contacto
Postcondición	El usuario se comunica con el servicio al cliente de manera efectiva

Ganchala L, Ocaña H. *Requerimiento Soporte Técnico; 2023*

ID RF-09	Búsqueda de productos
Actores	Cliente
Interfaz	Barra de búsqueda, filtros, categorías.
Precondición	El cliente debe tener acceso a la aplicación móvil.
Descripción	La aplicación debe permitir a los clientes buscar y encontrar productos específicos en la aplicación utilizando palabras clave, categorías, filtros y otros criterios de búsqueda.
Postcondición	El cliente encuentra los productos que está buscando y puede tomar una decisión informada sobre qué comprar.

Ganchala L, Ocaña H. *Requerimiento Búsqueda de productos; 2023*

ID RF-10	Geolocalización
Actores	Cliente
Interfaz	Mapa
Precondición	El usuario debe tener activado el GPS en su dispositivo y haber otorgado permisos a la aplicación para acceder a su ubicación
Descripción	La aplicación debe ser capaz de obtener la ubicación actual del usuario y mostrarla en un mapa
Postcondición	El usuario puede ver su ubicación actual en el mapa y utilizar esta información para otras funcionalidades de la aplicación

Ganchala L, Ocaña H. *Requerimiento Geolocalización*; 2023

4.1.3.2. Requerimientos no funcionales

Requerimientos No Funcionales
Seguridad
<ul style="list-style-type: none"> · El sistema debe proteger la información del usuario y las transacciones de manera segura, utilizando medidas de seguridad adecuadas. · Todas las comunicaciones externas entre servidores de datos y otros componentes de la aplicación deben estar cifradas. · La aplicación debe cumplir con las regulaciones relevantes de protección de datos y privacidad.
Rendimiento
<ul style="list-style-type: none"> · La aplicación debe ser rápida y responder a las acciones del usuario en un tiempo razonable. · Todas las funcionalidades de la aplicación, actividades y conexiones deben responder al usuario en menos de 5 segundos.

Usabilidad
<ul style="list-style-type: none"> · La aplicación debe ser fácil de usar y navegar, con una interfaz intuitiva y clara. · El tiempo de aprendizaje del sistema por parte de un usuario debe ser inferior a 1 hora. · La tasa de errores del usuario debe ser inferior al 1% en el sistema. · La aplicación debe contar con manuales de usuario estructurados adecuadamente. · La aplicación debe tener un módulo de ayuda y una interfaz gráfica bien diseñada.
Mantenibilidad
<ul style="list-style-type: none"> · La aplicación puede modificarse para corregir errores, mejorar su rendimiento, adaptarse a cambios en el entorno y agregar nuevas funcionalidades. · El tiempo promedio de duración de las fallas no debe superar los 15 minutos o se deben programar tiempos de mantenimiento anticipados. · La probabilidad de falla de la aplicación no debe ser superior al 0,05 · Se debe especificar un plan de recuperación ante desastres para la aplicación.
Escalabilidad
<ul style="list-style-type: none"> · La aplicación es escalable para agregar nuevas funcionalidades y requisitos sin afectar el código existente. · La aplicación debe tener una documentación adecuada para futuros cambios o integrantes del proyecto. · Se pueden incorporar aspectos de reutilización de código o reingeniería para garantizar la calidad del software.

Ganchala L, Ocaña H. *Requerimiento no funcionales* ; 2023

4.1.3.3. Requerimientos de interfaz

Requerimientos de Interfaz

- La aplicación debe mostrar los menús y productos en la pantalla de manera clara y atractiva.
- El usuario debe poder realizar un pedido de manera sencilla e intuitiva, seleccionando los productos que desea y especificando cualquier modificación o preferencia.
- La aplicación debe mostrar notificaciones y mensajes de error de manera clara y efectiva, informando al usuario sobre el estado de su pedido o cualquier problema que pueda surgir
- La aplicación debe presentar opciones de pago y entrega de manera clara y sencilla, permitiendo al usuario seleccionar la opción que mejor se adapte a sus necesidades.

Ganchala L, Ocaña H. *Requerimientos de interfaz*; 2023

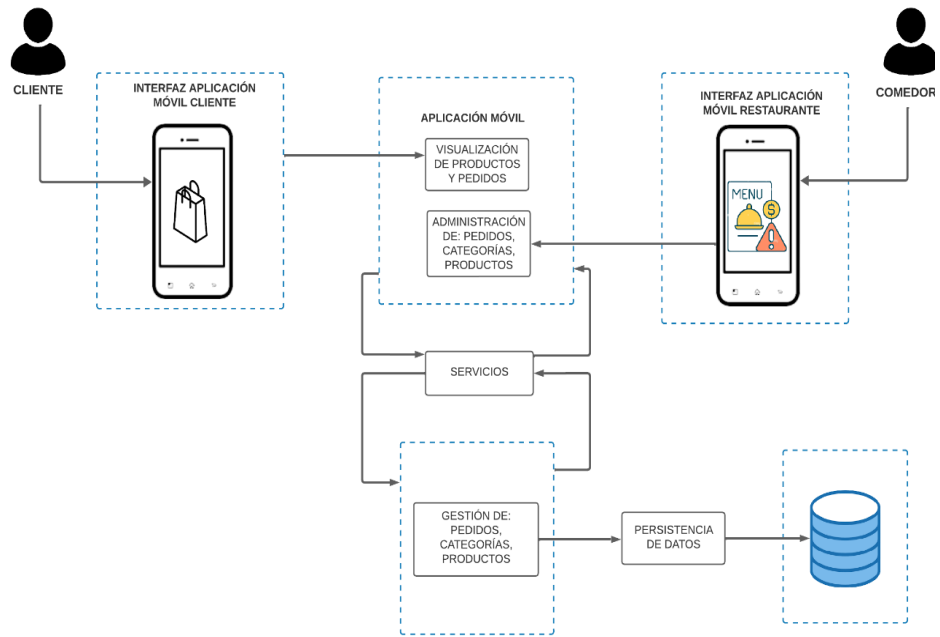
4.2. Diseño

4.2.1. Arquitectura del Software

La arquitectura de la aplicación móvil se basa en un enfoque cliente servidor, con una interfaz de usuario intuitiva, conectada a una base de datos que almacenara la información relevante de los locales, como el menú y los pedidos de los usuarios.

Figura 1

Diagrama de la arquitectura de aplicación móvil “Comedor UEB”



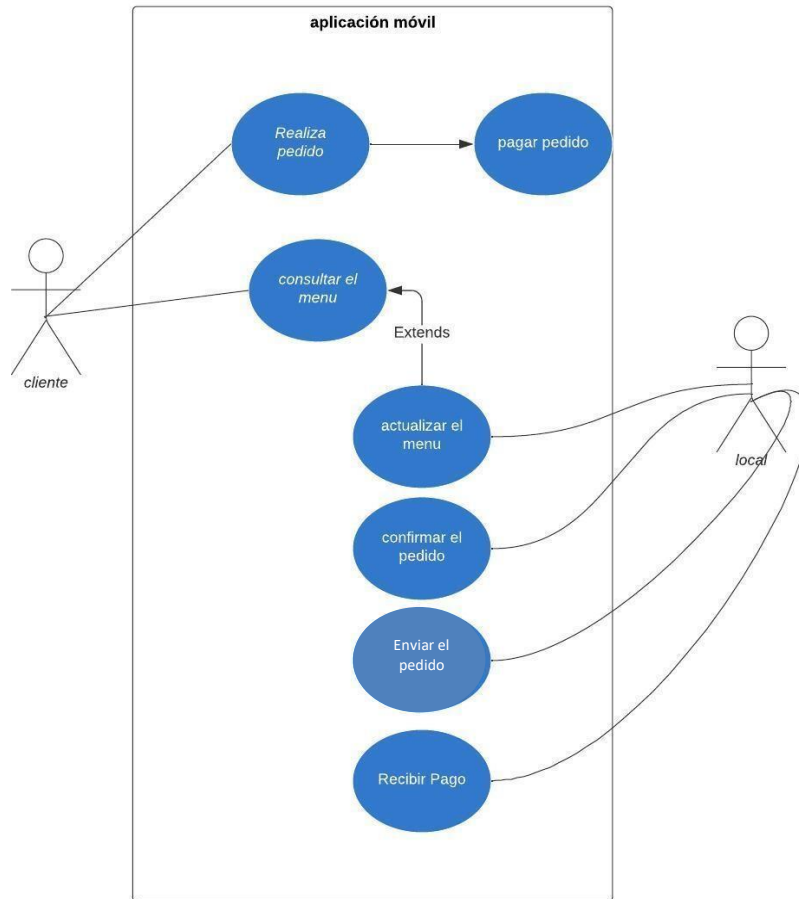
Ganchala L, Ocaña H. *Diagrama de arquitectura*; 2023

4.2.2. Diagramas

4.2.2.1. Diagrama de Caso de Uso

Figura 2

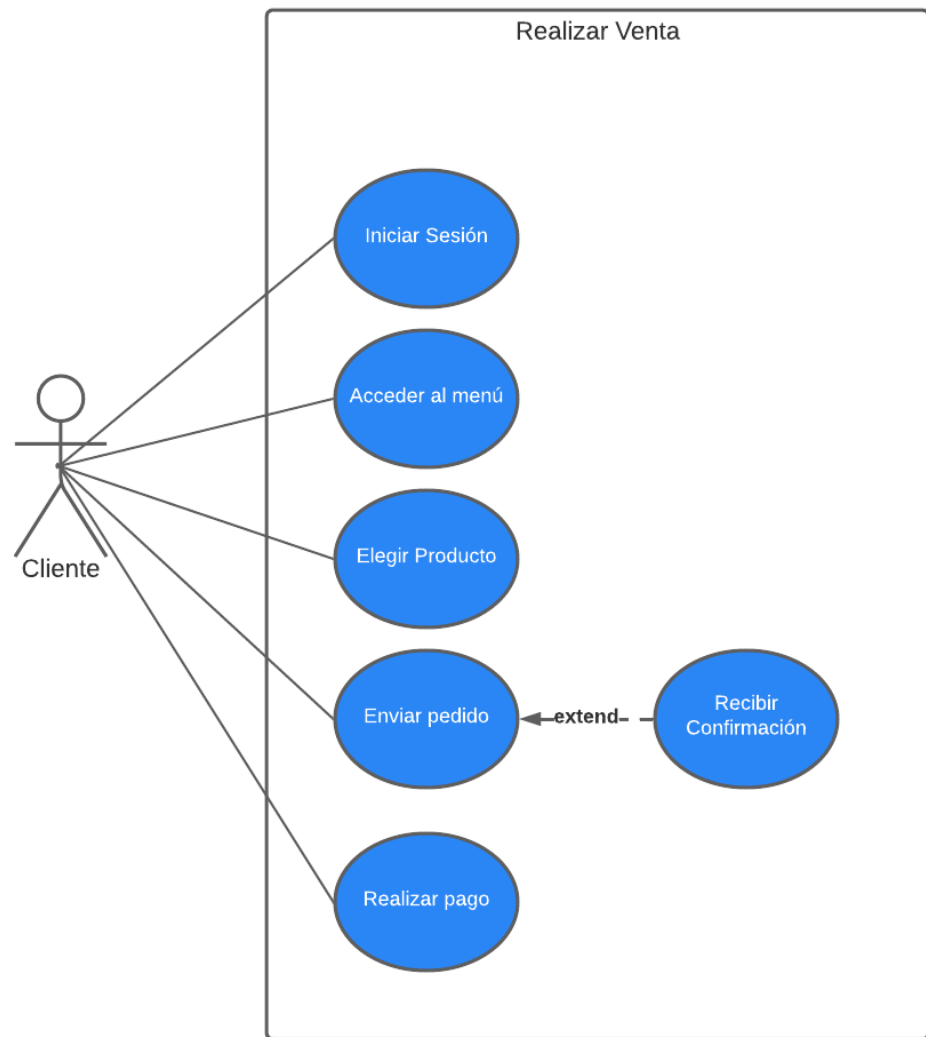
Diagrama general de la aplicación



Ganchala L, Ocaña H. *Diagrama*; 2023

Figura 3

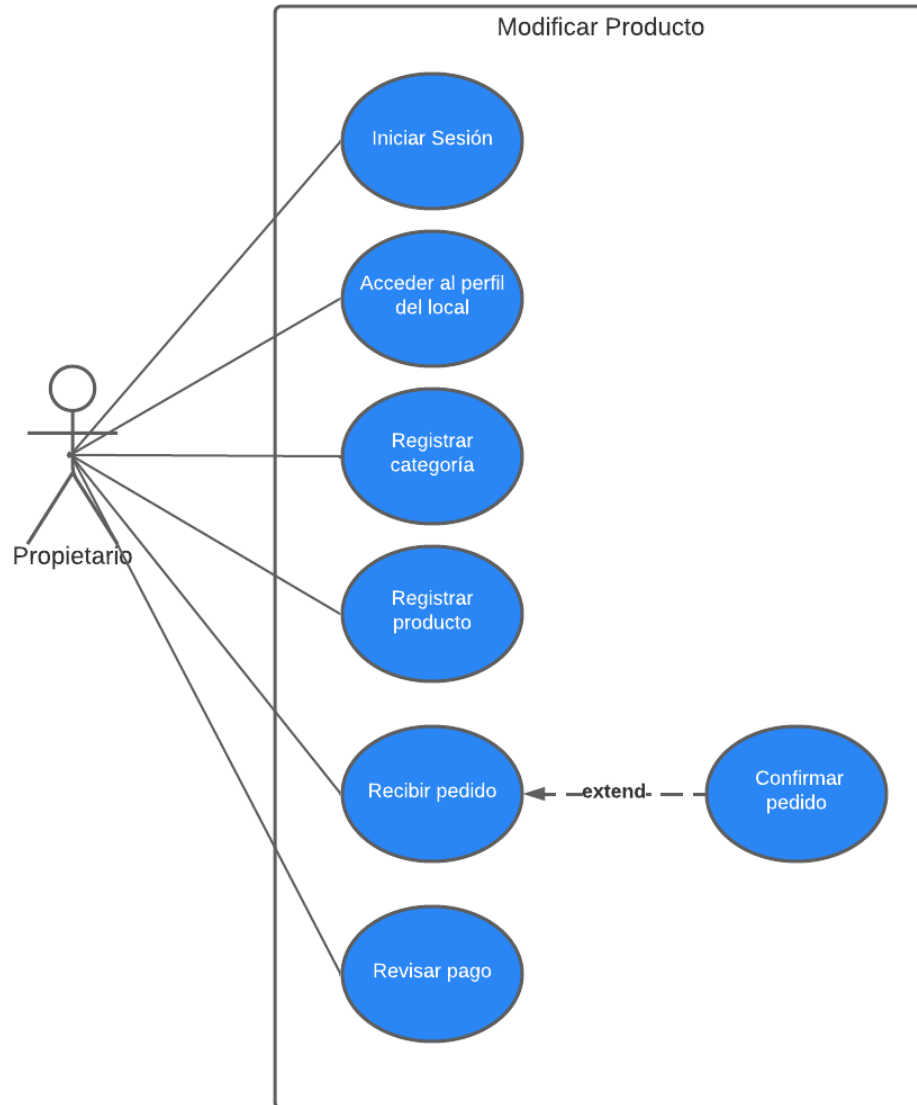
Diagrama de Caso de Uso Realizar Venta:



Ganchala L, Ocaña H. *Diagrama*; 2023

Figura 4

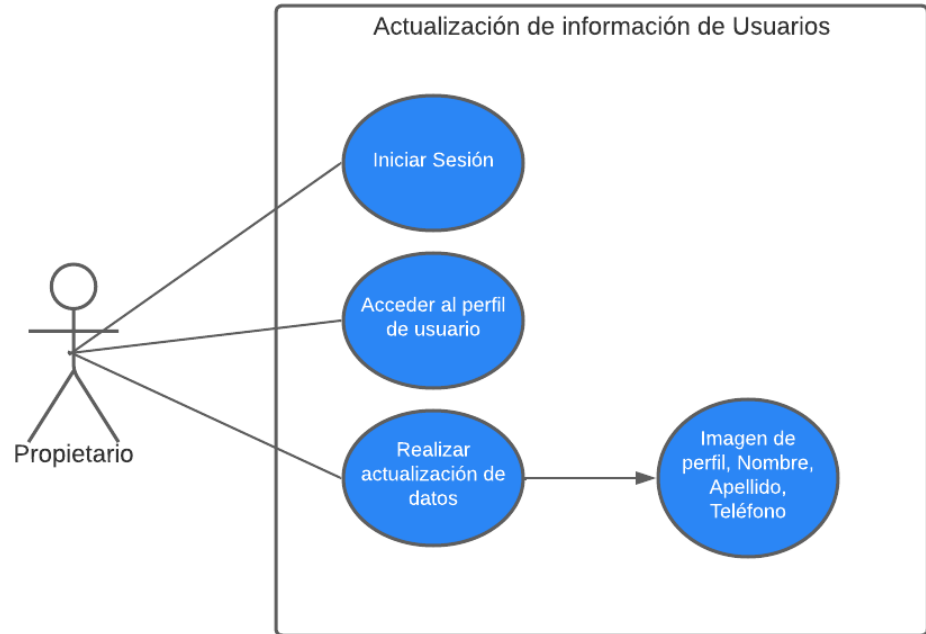
Diagrama de Caso de Uso Modificar Producto:



Ganchala L, Ocaña H. *Diagrama*; 2023.

Figura 5

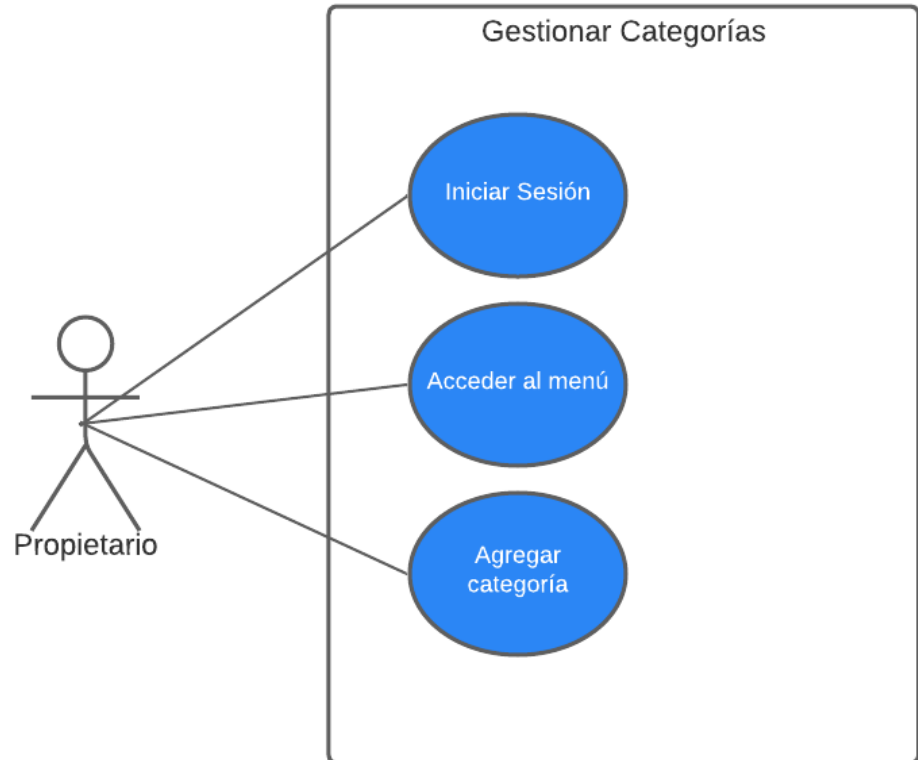
Diagrama de Caso de Uso Gestionar Usuario:



Ganchala L, Ocaña H. *Diagrama*; 2023.

Figura 6

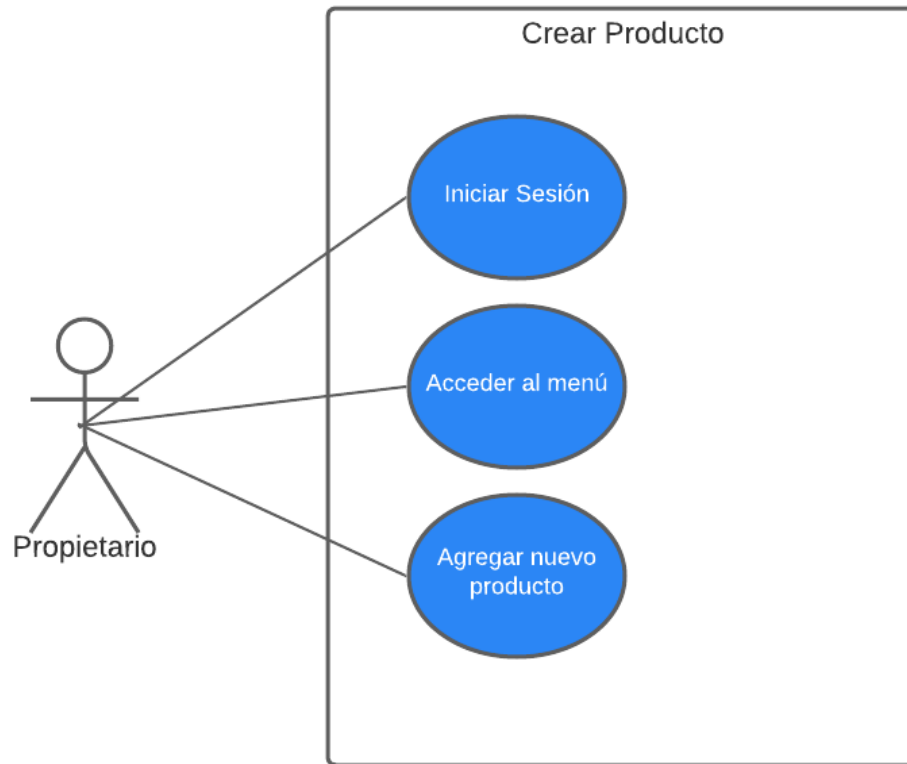
Diagrama de Caso de Uso Gestionar Categorías:



Ganchala L, Ocaña H. *Diagrama*; 2023

Figura 7

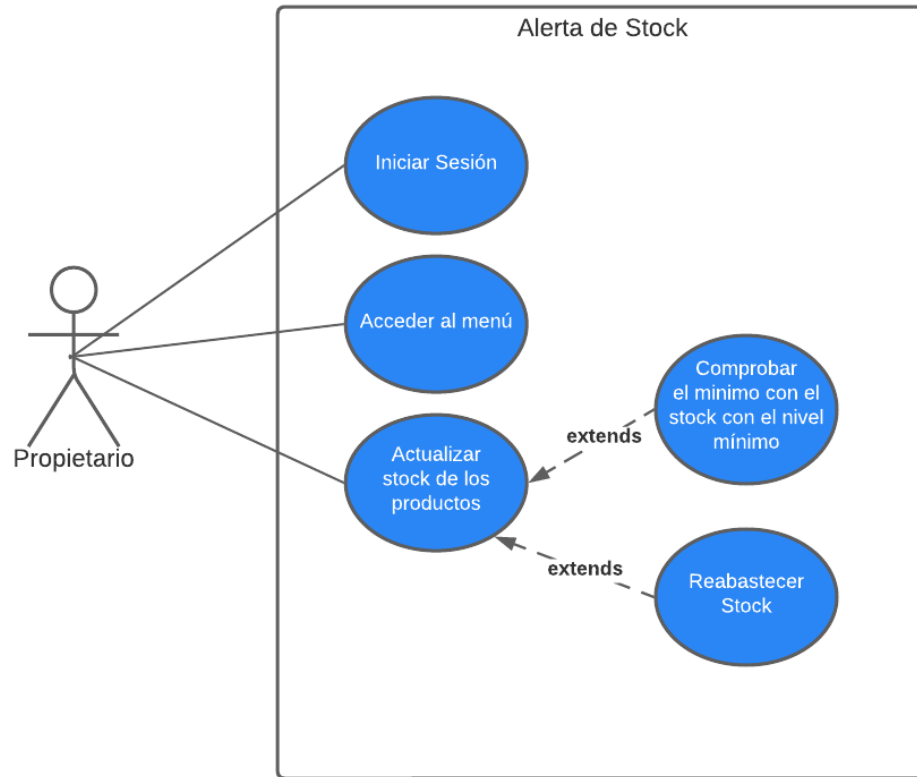
Diagrama de Caso de Uso Crear Producto:



Ganchala L, Ocaña H. *Diagrama*; 2023.

Figura 8

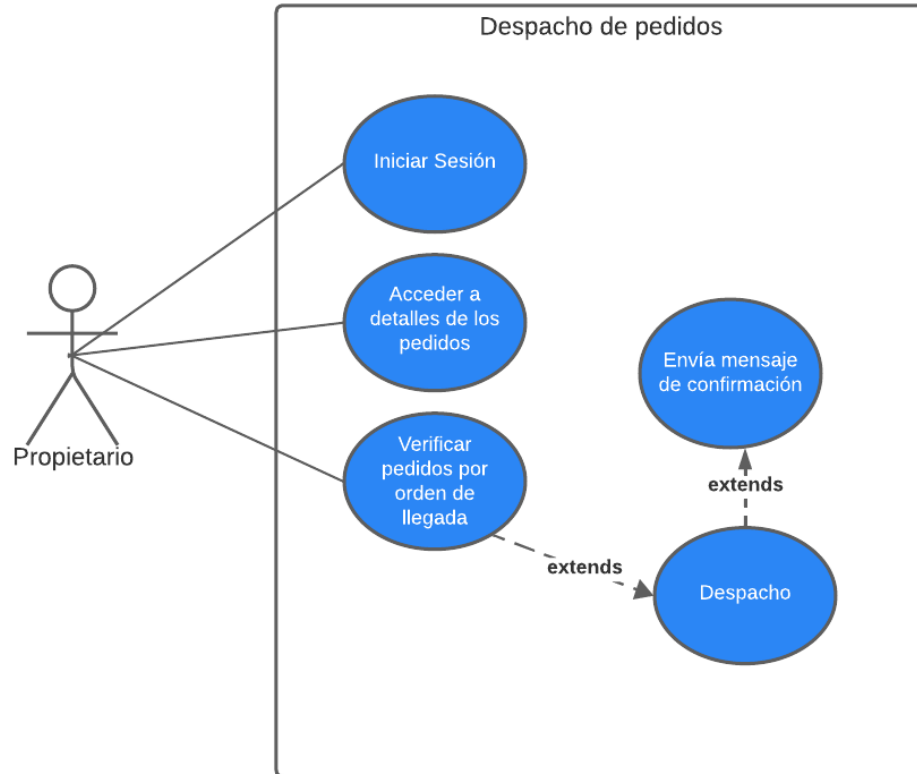
Diagrama de Caso de Uso Generar Alerta de Stock:



Ganchala L, Ocaña H. *Diagrama*; 2023.

Figura 9

Diagrama de Caso de Uso Despacho de Pedidos

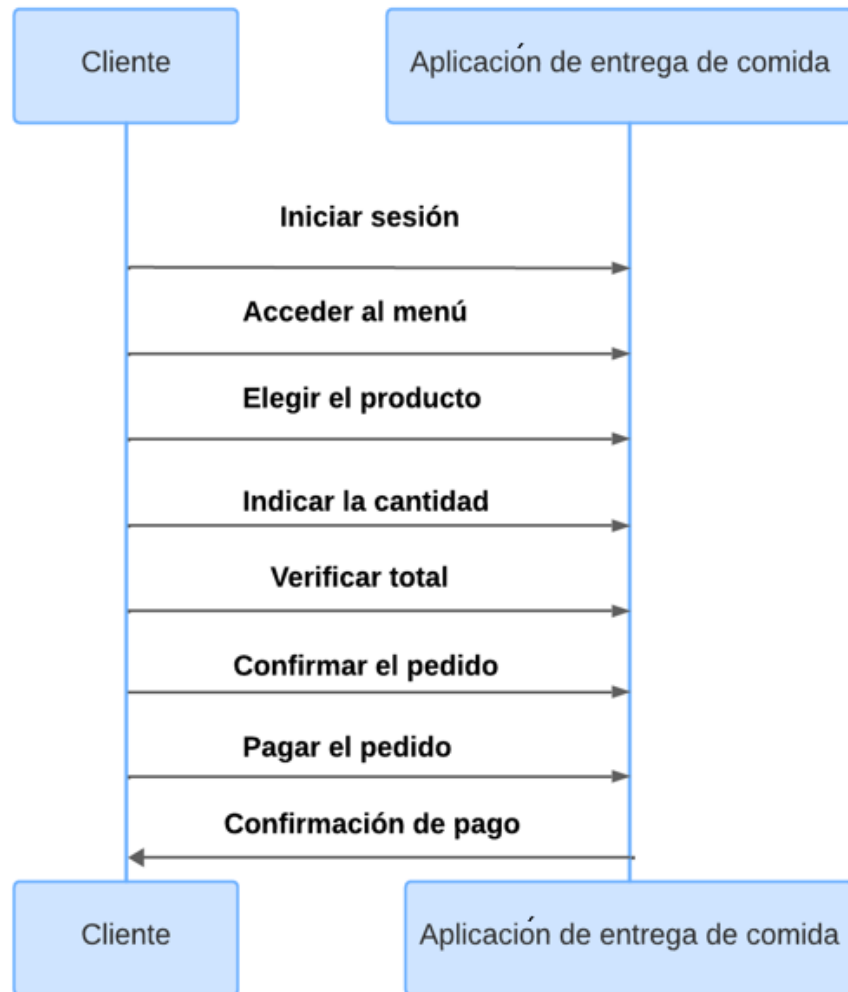


Ganchala L, Ocaña H. *Diagrama*; 2023.

4.2.2.2. Diagrama de Secuencia

Figura 10

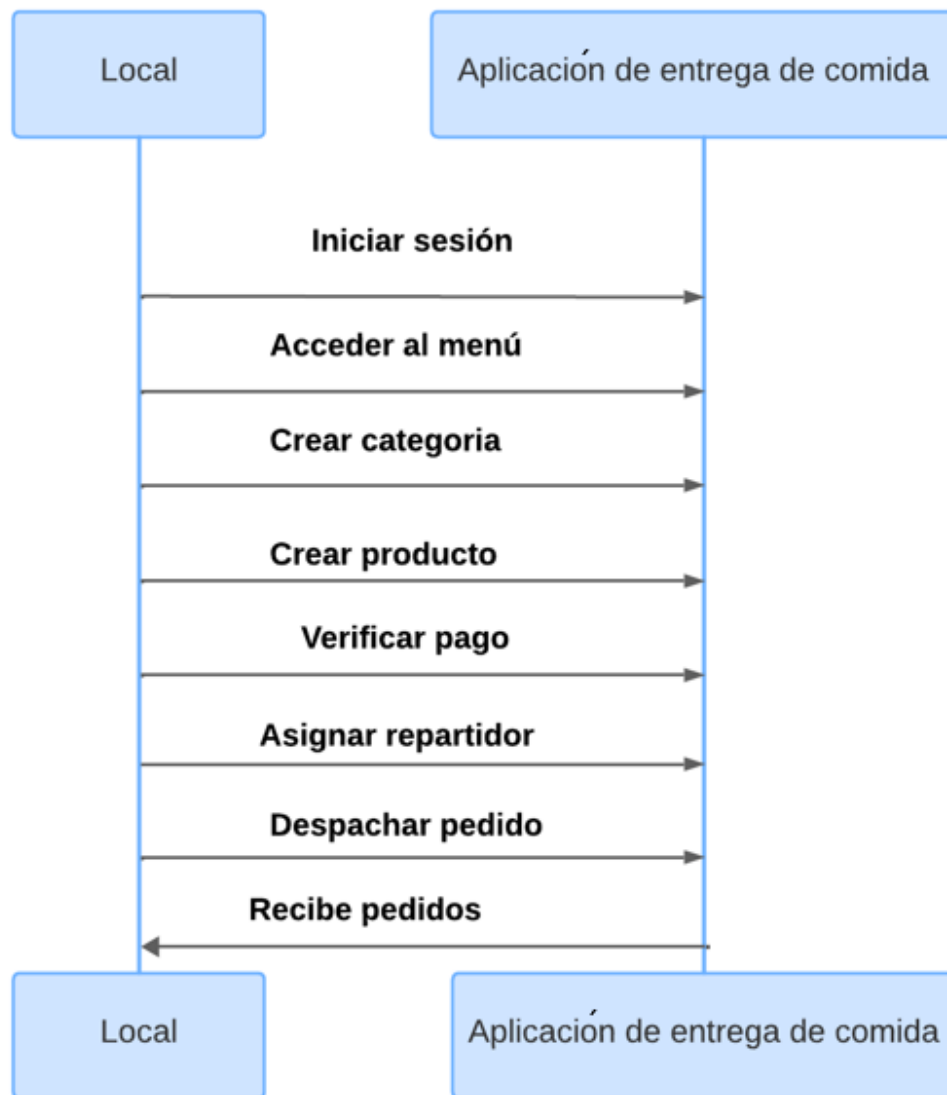
Diagrama de secuencia cliente



Ganchala L, Ocaña H. *Diagrama de secuencia*; 2023.

Figura 11

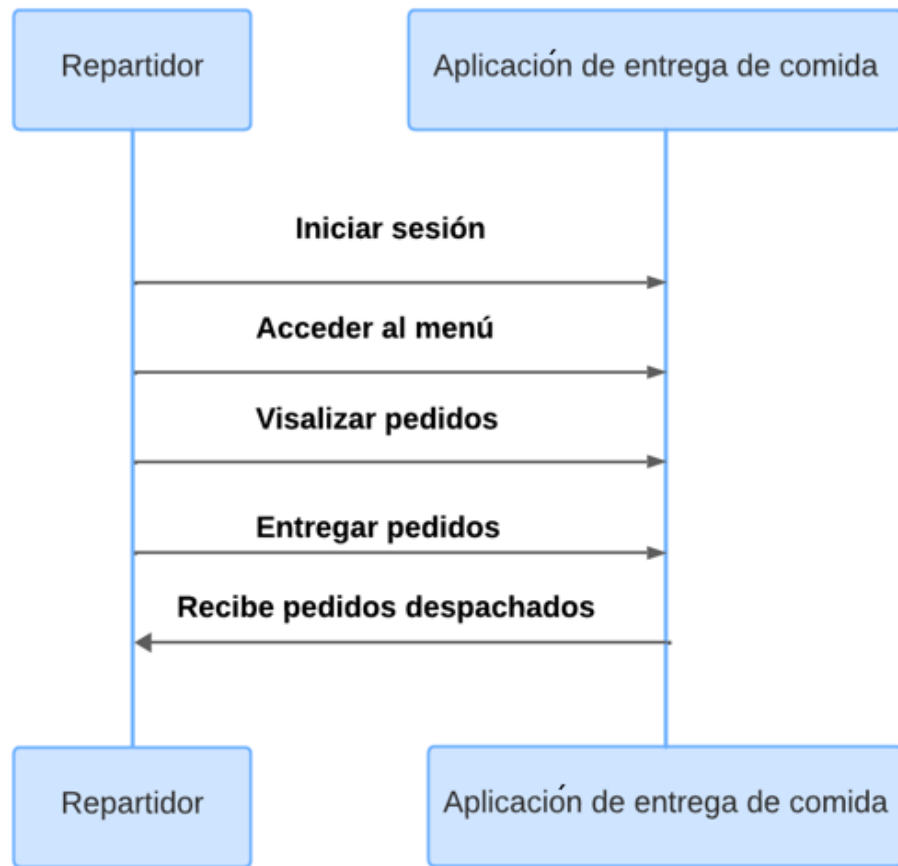
Diagrama de secuencia local



Ganchala L, Ocaña H. *Diagrama de secuencia del local*; 2023.

Figura 12

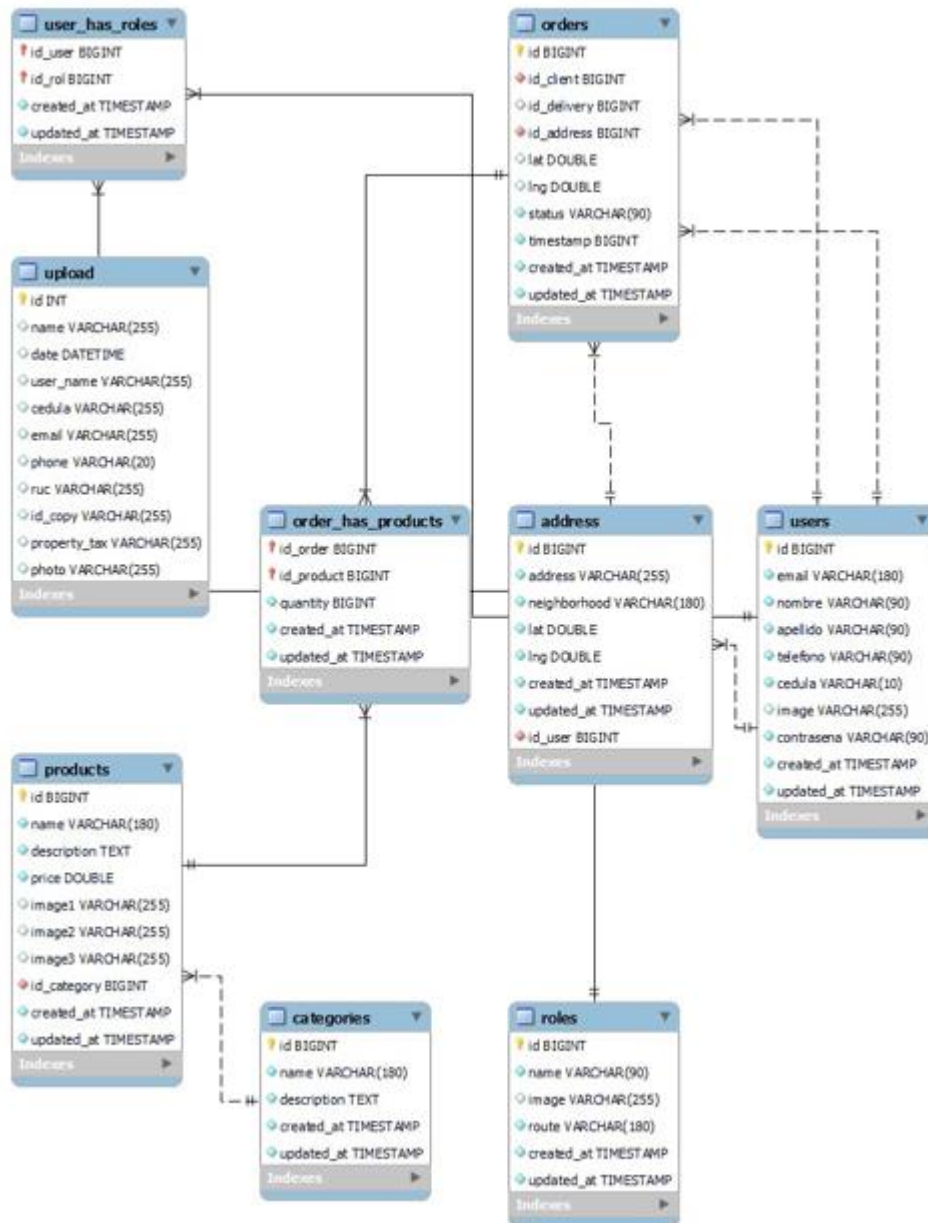
Diagrama de Secuencia repartidor



Ganchala L, Ocaña H. *Diagrama de secuencia del repartidor*; 2023.

4.2.2.3. Diagrama Entidad Relación

Figura 13



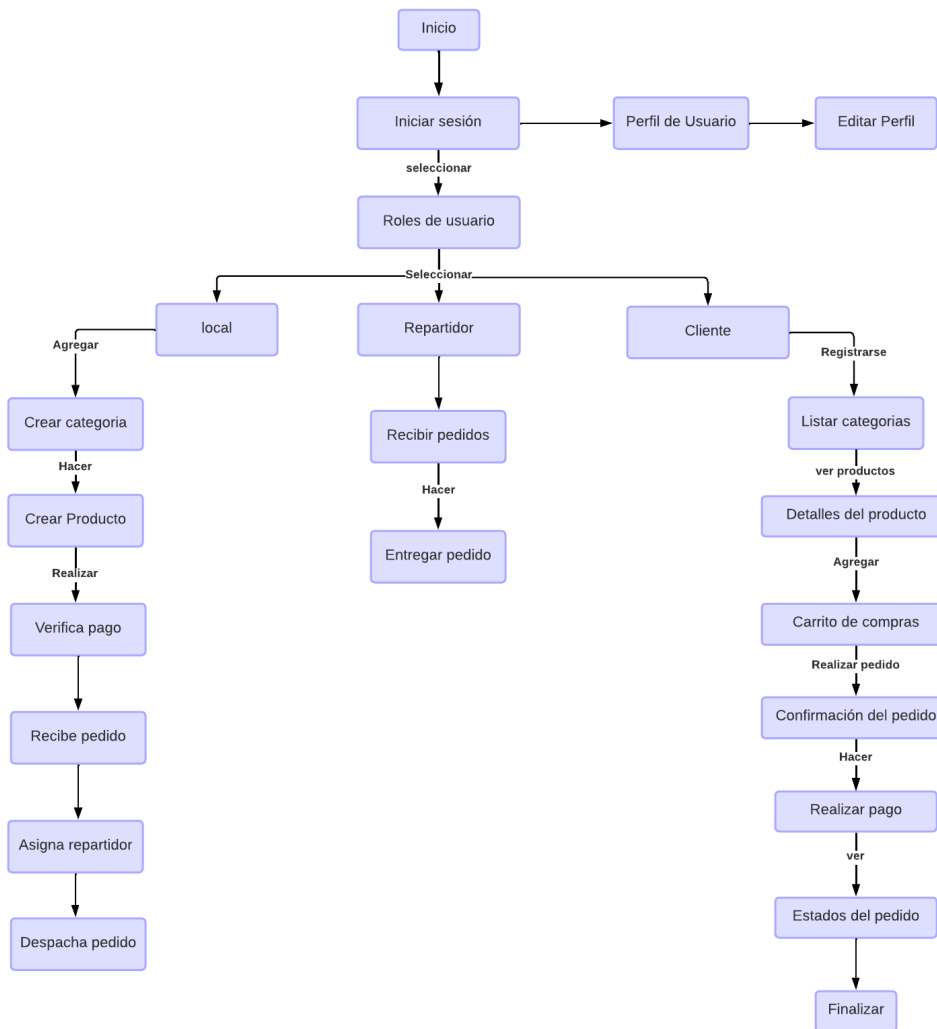
Ganchala L, Ocaña H. *Diagrama entidad relación*; 2023.

4.2.3. Diseño de la Interfaz

4.2.3.1. Patrón de navegación

Figura 14

Diagrama de Patrón de navegación



Ganchala L, Ocaña H. *Diagrama*; 2023.

4.2.3.2. Interfaz de Pantallas

Color: En el diseño de la aplicación móvil “Comedor UEB” se escogió un tono rojo, porque está asociado al estímulo del apetito, captando la atención del consumidor.

Los colores establecidos en la aplicación son:

Como primario el color rojo

Figura 15



Como secundario el color azul

Figura 16



Y una combinación de gris.

Figura 17



Tipografía: La tipografía utilizada en la aplicación cumplen con los atributos visuales de la misma.

Tipo de letra: Roboto

Tamaño: 150 píxeles de ancho y 150 píxeles de altura.

Logo: En el logo se puede apreciar las iniciales de la Universidad Estatal de Bolívar, en el cual se le adornó con un sombrero de chef y unos cubiertos que

representan los servicios de alimentación, encerrado en una circunferencia de una tonalidad naranja dorado que representa una sensación de calidez y apetito. El menú principal se encuentra ubicado en la parte inferior de la ventana y cuenta con cuatro opciones pedidos, categoría, producto, perfil

Figura 18

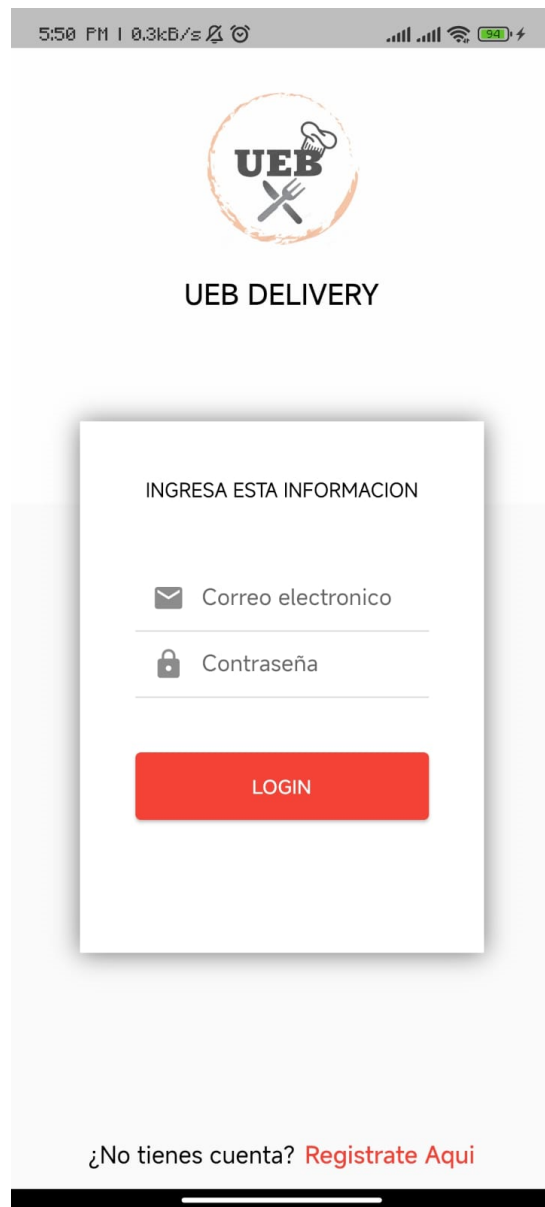
Logo del comedor



Ganchala L, Ocaña H. *Icono Del Programa.*; 2023.

Figura 19

Pantalla de Inicio de sesión

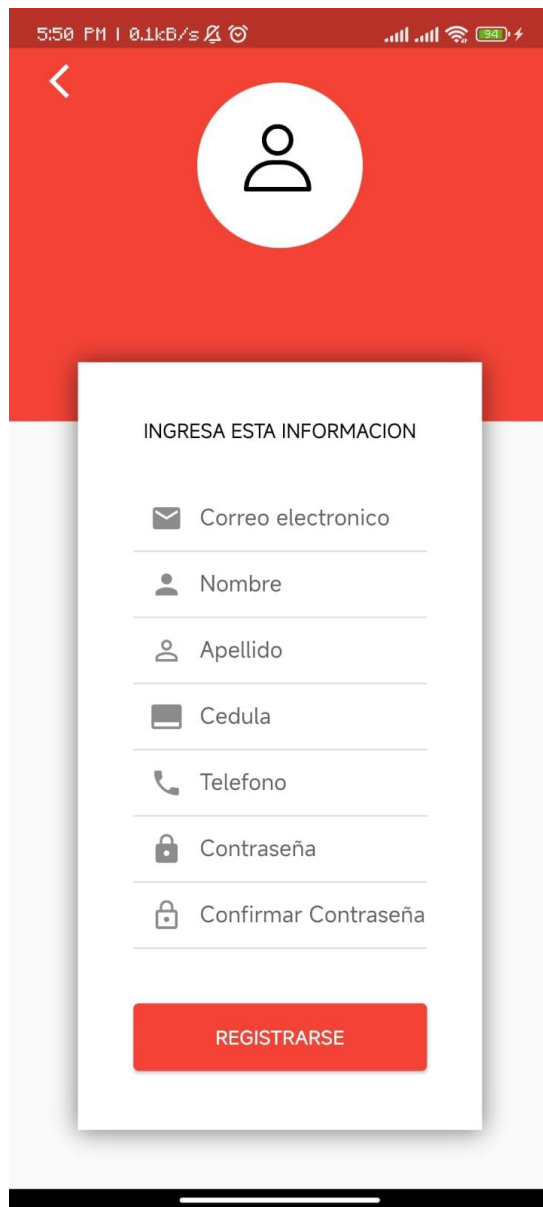


Ganchala L, Ocaña H. *App Test.*; 2023.

En la pantalla de inicio de sesión le permite al usuario ingresar el correo y contraseña para ingresar a los servicios de aplicación

Figura 20

Pantalla de Registro

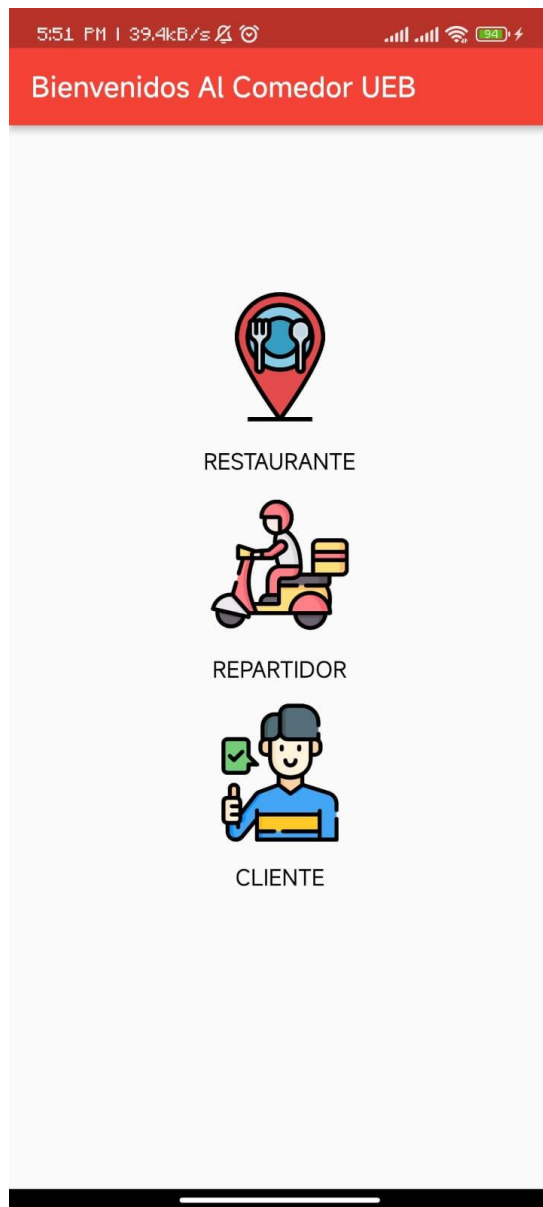


Ganchala L, Ocaña H. *App Test.*; 2023.

En la pantalla de registro le permite al cliente ingresar sus datos personales para poder crear la cuenta y poder acceder a los servicios de alimentación

Figura 21

Pantalla de Roles

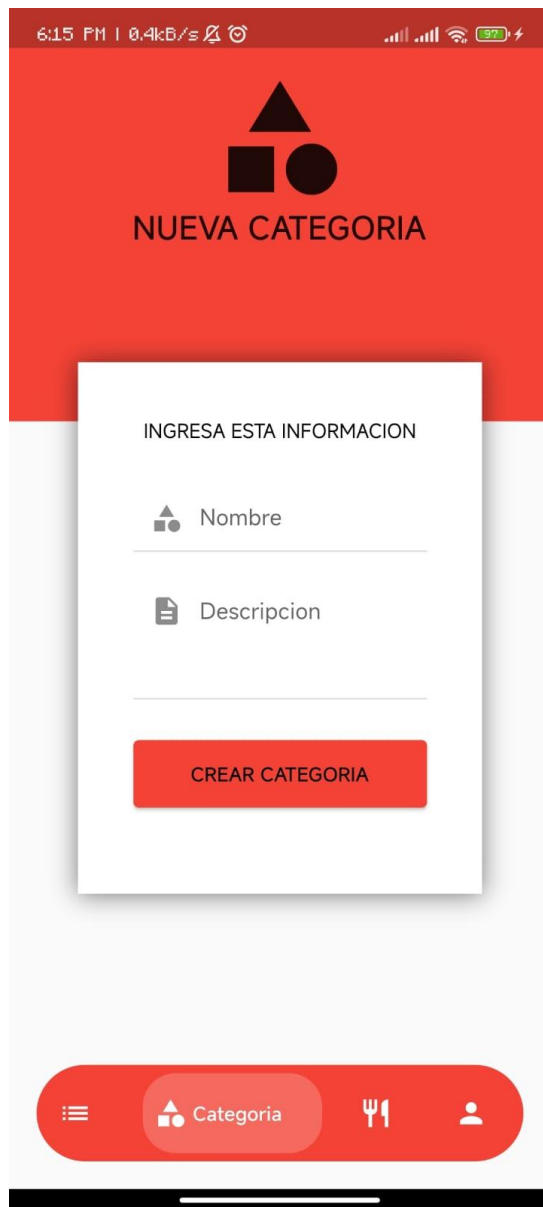


Ganchala L, Ocaña H. *App Test.*; 2023.

En esta sección el usuario puede escoger un rol, en el apartado de restaurante puede ingresar los servicios de alimentación, el apartado de repartidor para visualizar las órdenes a entregar

Figura 22

Pantalla de Crear nueva categoría

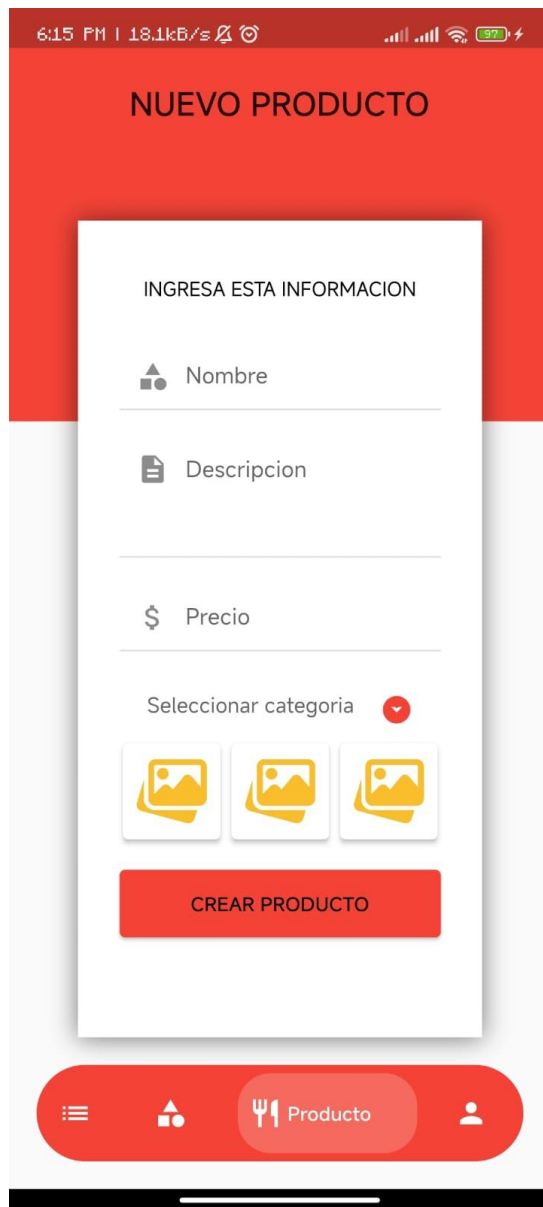


Ganchala L, Ocaña H. *App Test.*; 2023.

El apartado de restaurante: la sección categoría le permite crear las categorías que puede ofrecer en su local, ingresando la información correspondiente.

Figura 23

Pantalla Crear nuevo producto

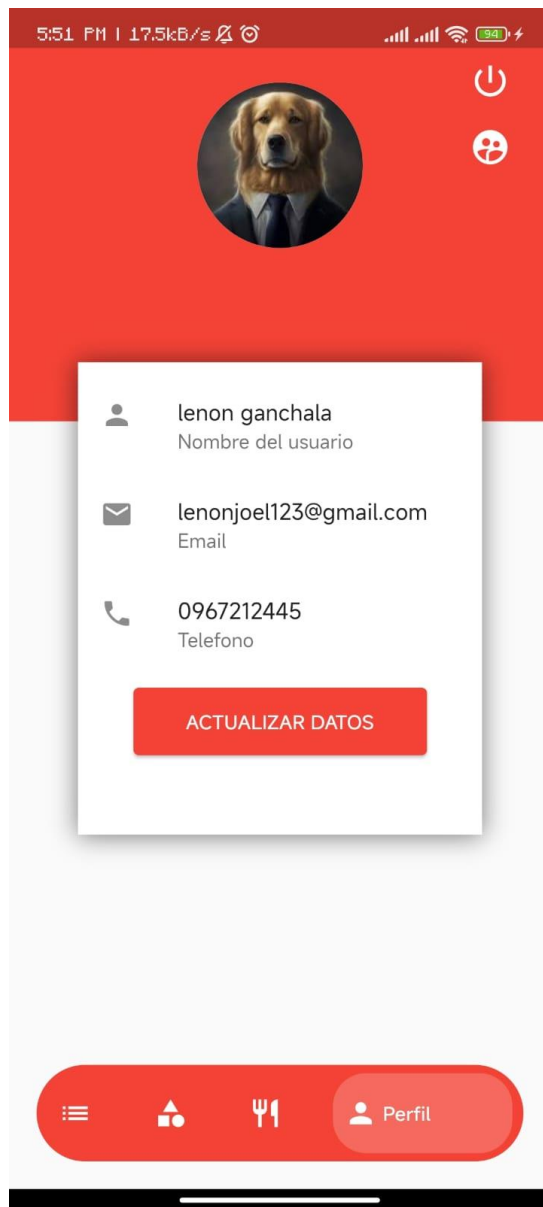


Ganchala L, Ocaña H. *App Test.*; 2023.

El apartado de restaurante: la sección producto le permite ingresar la información necesaria de sus productos, seleccionando las categorías que posee y las imágenes de sus productos.

Figura 24

Pantalla de perfil de usuario

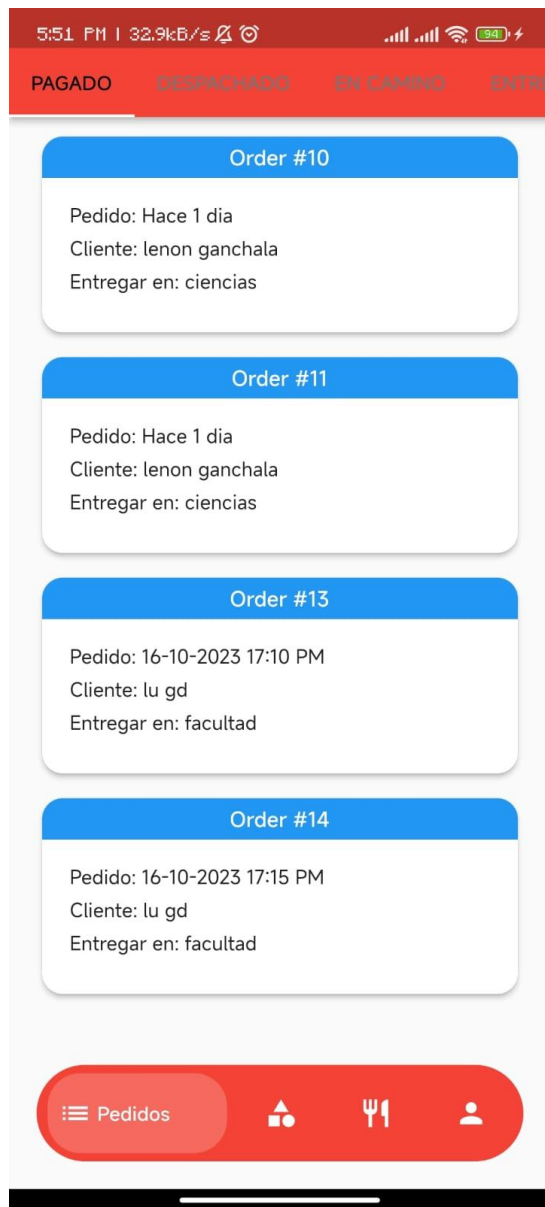


Ganchala L, Ocaña H. *App Test.*; 2023.

El apartado de restaurante: la sección de perfil muestra la información del usuario para que este pueda actualizarlo si fuese necesario.

Figura 25

Pantalla de pedidos

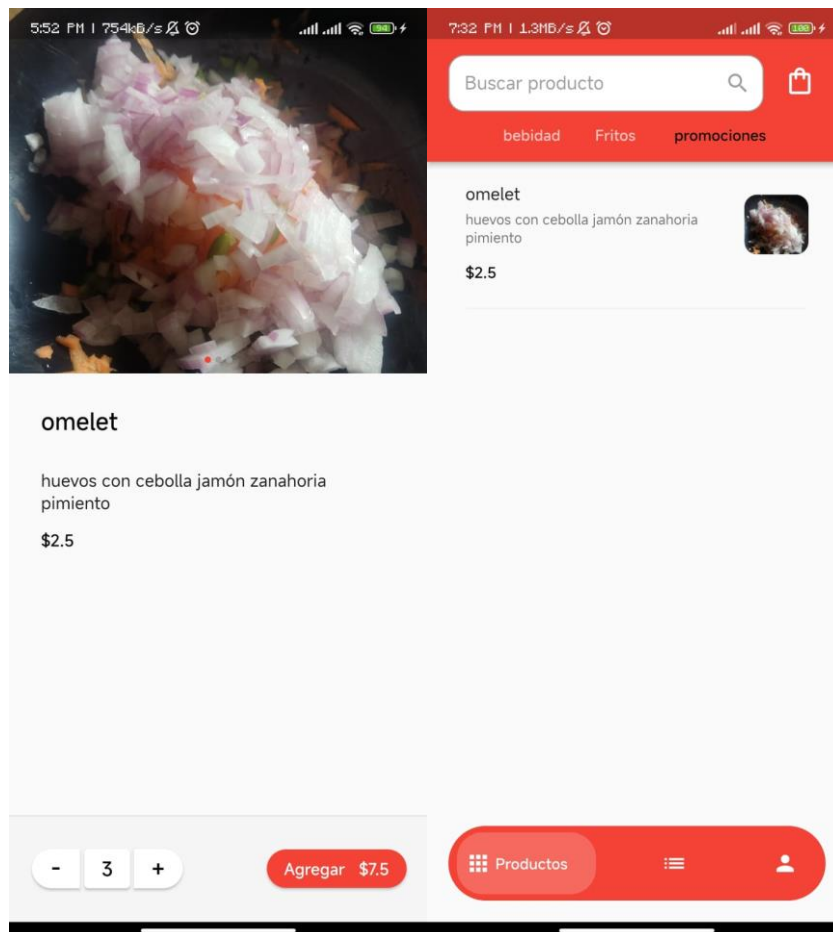


Ganchala L, Ocaña H. *App Test.*; 2023.

El apartado de restaurante: la sección de pedidos le permite al usuario visualizar los pedidos confirmados para prepararlos.

Figura 26

Seleccionar y agregar productos

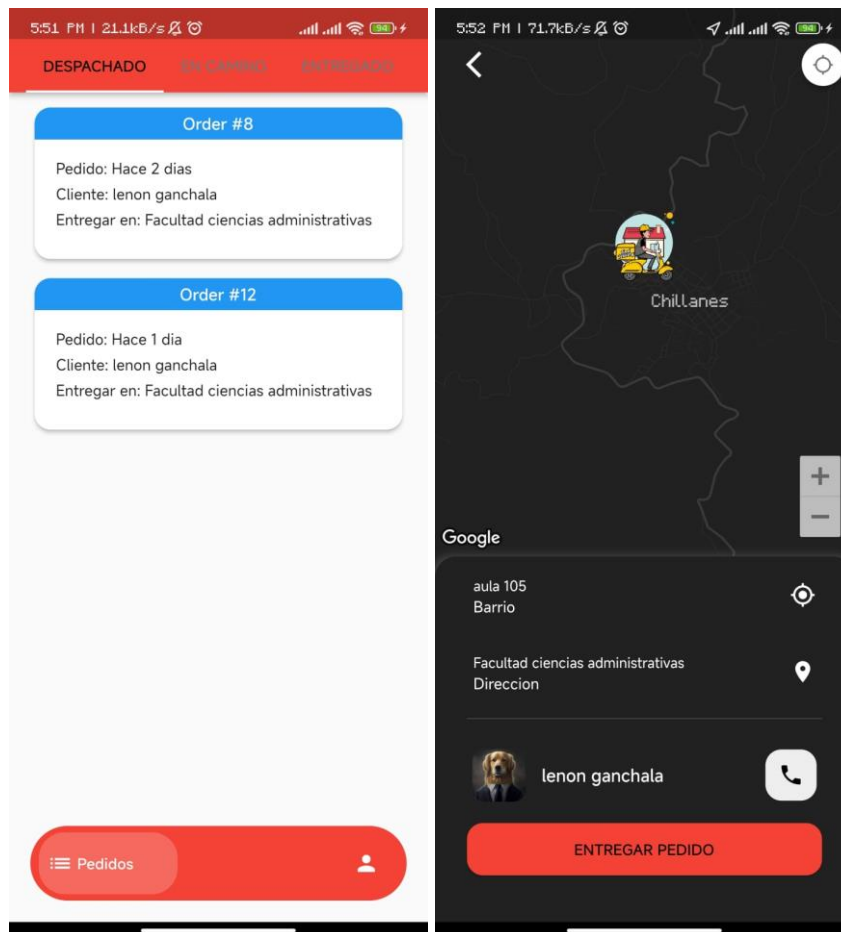


Ganchala L, Ocaña H. *App Test.*; 2023.

En la sección de productos le permite al cliente visualizar los productos y las categorías, que al momento de seleccionar un producto se despliega una pantalla en la que puede modificar la cantidad y agregar al carrito

Figura 27

Pantalla de pedidos a despachar

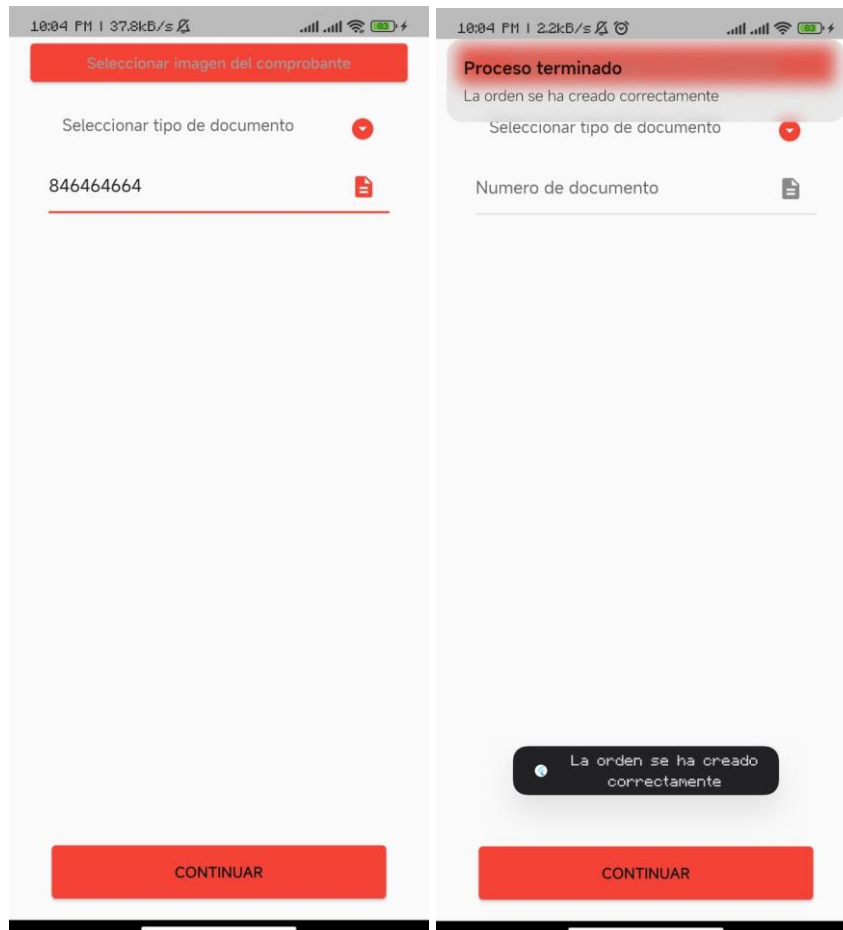


Ganchala L, Ocaña H. *App Test.*; 2023.

El apartado de repartidor: en esta sección le permite al usuario visualizar los pedidos que tiene que entregar los cuales pueden pasar por varios estados en despachado, encamino, entregado, además puede dirigirse por la geolocalización

Figura 28

Pantalla de pagos

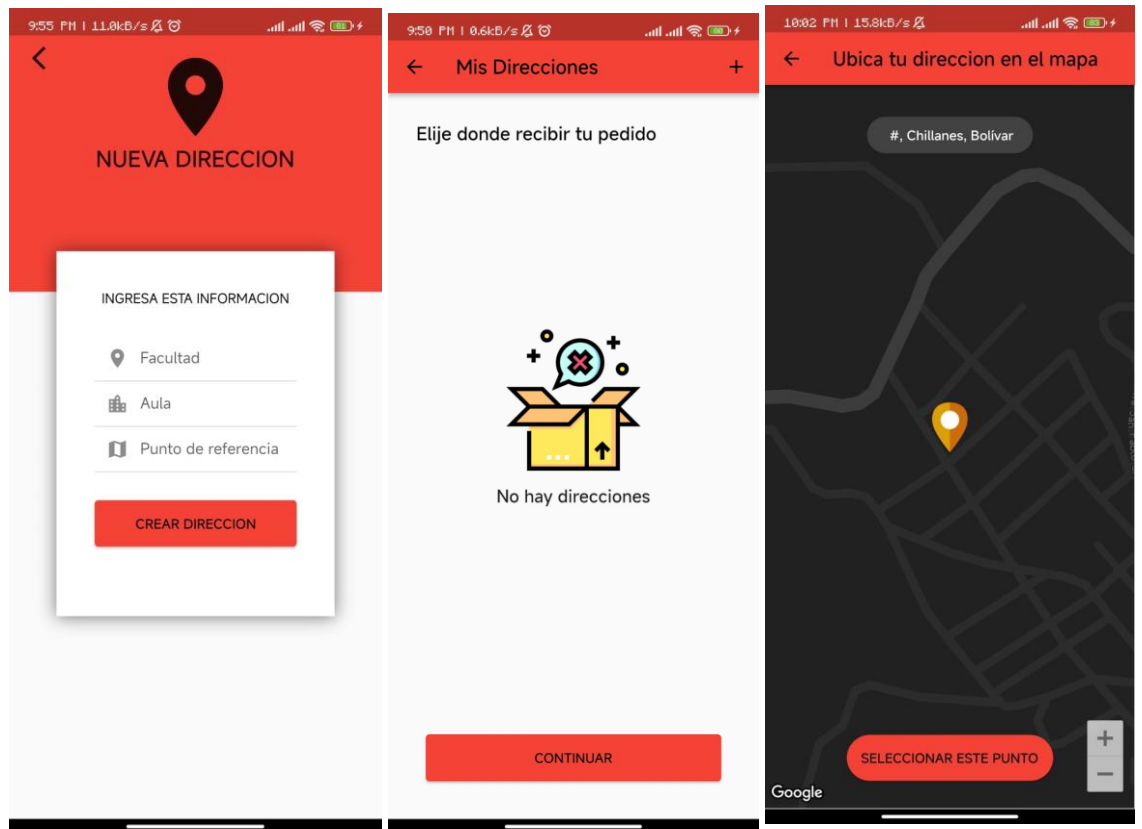


Ganchala L, Ocaña H. *App Test.*; 2023.

En esta sección le permite al cliente seleccionar el documento que en este caso sería la imagen del comprobante de pago y presionar el boto de continuar para que se envíe y se confirme la orden.

Figura 29

Pantalla de direcciones



Ganchala L, Ocaña H. *App Test.*; 2023.

En esta sección le permite al cliente agregar una nueva dirección además de poder agregar la ubicación en el mapa para que sea eficiente la entrega del pedido.

4.3.Programación

4.3.1. Definición del objetivo

Desarrollar una aplicación móvil, permitiendo a los usuarios adquirir comida de forma eficiente y seleccionando el método de pago disponible en los diferentes locales del comedor de la Universidad Estatal de Bolívar.

4.3.2. Análisis del problema

Los servicios de alimentación que se brindan en los locales del comedor de la Universidad Estatal de Bolívar no cuentan con una gestión eficiente de pedidos, por lo cual la atención al cliente es lenta, además la falta de actualización del menú y la disponibilidad de los alimentos se deben a que suelen ser escritos en pizarras y promocionados en cuanto el cliente llega al comedor, esta mala gestión de la información hacen que el cliente no se decida sobre qué local visitar, afectando de forma negativa la experiencia del cliente resultando en un alejamiento del servicio que quieren proporcionar.

Por lo cual se desarrolló una aplicación móvil que mejora el proceso de los pedidos, optimizando los tiempos de atención, dando a los clientes acceso a un menú completo de cada local del comedor, en el cual se da a conocer la disponibilidad de los alimentos en tiempo real. Además, permite al cliente realizar pedidos y escoger el método de pago disponible, mejorando de esta forma la experiencia del cliente.

4.3.3. Diseño del Algoritmo

Apartado del propietario del local (

Módulo de inicio de sesión

1. Inicio
2. Seleccionar módulo de inicio de sesión
3. Ingresar datos de inicio de sesión (correo, contraseña)
4. Si los datos son correctos entonces
 - 4.1. Inicia sesión y redirige al usuario a la página principal
 - 4.2. Fin

Módulo Crear Nueva Categoría

1. Inicio
2. Seleccionar módulo crear categoría
3. Ingresar nombre de la categoría
4. Ingresar descripción de la categoría
5. Si los datos son correctos entonces
 - 5.1. Crear nueva categoría
 - 5.2. Mostrar mensaje de éxito
 - 5.3. Fin
6. Sino
 - 6.1. Mostrar mensaje de error y volver al paso 3

Módulo Crear Nuevo Producto

1. Inicio
2. Seleccionar módulo nuevo producto
3. Ingresar nombre
4. Ingresar descripción
5. Ingresar precio
6. Seleccionar categoría
7. Seleccionar 3 imágenes
8. Si los datos son correctos entonces
 - 8.1. Crear nuevo producto
 - 8.2. Mostrar mensaje de éxito
 - 8.3. Fin
9. Sino
 10. Mostrar mensaje de error y volver al paso 3

)

Apartado de Clientes (

Registro de usuarios

5. Inicio
6. Seleccionar módulo de registro
7. Ingresar datos de registro (seleccionar imagen de perfil, correo, nombre, apellido, cédula, teléfono, contraseña)

8. Si los datos son correctos entonces
 - 4.1. Crear cuenta de usuario
 - 4.2. Mostrar mensaje de éxito
 - 4.3. Fin
 - 4.4. Si no
 - 4.5. Mostrar mensaje de error de formulario
 - 4.5.1. Volver al paso 3
 - 4.6. Fin

Inicio de sesión

1. Inicio
2. Módulo de inicio de sesión
3. Ingresar datos de usuario creado (correo, contraseña)
4. Si los datos son correctos entonces
 - 4.1. Iniciar sesión
 - 4.2. Redirigir al usuario al apartado de cliente
 - 4.3. Fin
5. Sino
 - 5.1. Mostrar mensaje de error
 - 5.2. Volver al paso 3

Módulo de Visualización de productos

1. Inicio
2. Seleccionar el módulo de productos
3. Mostrar lista de productos disponibles
4. Seleccionar producto
5. Mostrar detalle del producto
6. Si el usuario desea comprar entonces
 - 6.1. Ingresar cantidad deseada
 - 6.2. Añadir producto al carrito
 - 6.3. Si el usuario desea comprar más entonces
 - 6.4. Volver al paso 3
 - 6.5. Sino
 - 6.6. Proceder al módulo de realización de pedidos

7. Sino

7.1.Volver al paso 3

Módulo de Realización de Pedidos

1. Inicio
 2. Mostrar carrito de compras
 3. Si el usuario desea realizar pedido entonces
 - 3.1.Seleccionar método de pago
 - 3.2.Confirmar dirección de entrega
 - 3.3.Confirmar pedido
 - 3.4.Si la confirmación fue exitosa entonces
 - 3.4.1. Mostrar mensaje de éxito
 - 3.4.2. Fin
 - 3.5.Sino
 - 3.5.1.Mostrar mensaje de error
 - 3.5.2. Volver al paso 3
-)

Módulo de pagos (

1. Inicio
 2. Ingresar a módulo de pagos
 3. Seleccionar subir comprobante
 4. Seleccionar imagen
 5. Enviar
 6. Fin
 7. Sino
 8. Mostrar mensaje de error
 9. Regresar paso 3
-)

4.3.4. Codificación

Crear orden

```
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:udemy_flutter_delivery/src/models/product.dart';
import 'package:udemy_flutter_delivery/src/pages/client/orders/create/client_orders_create_controller.dart';
import 'package:udemy_flutter_delivery/src/widgets/no_data_widget.dart';

class ClientOrdersCreatePage extends StatelessWidget {
  ClientOrdersCreateController con = Get.put(ClientOrdersCreateController());

  @override
  Widget build(BuildContext context) {
    return Obx(() => Scaffold(
      bottomNavigationBar: Container(
        color: Color.fromRGBO(245, 245, 245, 1),
        height: 100,
        child: _totalToPay(context),
      ),
      appBar: AppBar(
        iconTheme: IconThemeData(
          color: Colors.black
        ),
        title: Text(
          'Mi Orden',
          style: TextStyle(
            color: Colors.black
          ),
        ),
      ),
      body: con.selectedProducts.length > 0
        ? ListView(
            children: con.selectedProducts.map((Product product) {
              return _cardProduct(product);
            }).toList(),
          )
        : Center(
            child: NoDataWidget(text: 'No hay ningun producto agregado aun')
          ),
    ));
  }

  Widget _totalToPay(BuildContext context) {
    return Column(
      children: [
        Divider(height: 1, color: Colors.grey[300]),
        Container(
          margin: EdgeInsets.only(left: 20, top: 25),
          child: Row(
            mainAxisAlignment: MainAxisAlignment.center,
            children: [
              Text(
```

```

        "TOTAL: \${con.total.value}',
        style: TextStyle(
          fontWeight: FontWeight.bold,
          fontSize: 17
        ),
      ),
    ),
    Container(
      margin: EdgeInsets.symmetric(horizontal: 30),

      child: ElevatedButton(
        onPressed: () => con.goToAddressList(),
        style: ElevatedButton.styleFrom(
          padding: EdgeInsets.all(15)
        ),
        child: Text(
          'CONFIRMAR ORDER',
          style: TextStyle(
            color: Colors.black
          ),
        ),
      ),
    ),
  ],
),
)
],
);
}

Widget _cardProduct(Product product) {
  return Container(
    margin: EdgeInsets.symmetric(horizontal: 20, vertical: 10),
    child: Row(
      children: [
        _imageProduct(product),
        SizedBox(width: 15),
        Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            Text(
              product.name ?? "",
              style: TextStyle(
                fontWeight: FontWeight.bold
              ),
            ),
            SizedBox(height: 7),
            _buttonsAddOrRemove(product)
          ],
        ),
        Spacer(),
        Column(
          children: [
            _textPrice(product),
            _iconDelete(product)
          ],
        ),
      ],
    ),
  ],
);
}

```



```

    ),
  );
}

Widget _iconDelete(Product product) {
  return IconButton(
    onPressed: () => con.deleteItem(product),
    icon: Icons.delete,
    color: Colors.red,
  );
}

Widget _textPrice(Product product) {
  return Container(
    margin: EdgeInsets.only(top: 10),
    child: Text(
      '\${product.price! * product.quantity!}',
      style: TextStyle(
        color: Colors.grey,
        fontWeight: FontWeight.bold
      ),
    ),
  );
}

Widget _buttonsAddOrRemove(Product product) {
  return Row(
    children: [
      GestureDetector(
        onTap: () => con.removeItem(product),
        child: Container(
          padding: EdgeInsets.symmetric(horizontal: 12, vertical: 7),
          decoration: BoxDecoration(
            color: Colors.grey[200],
            borderRadius: BorderRadius.only(
              topLeft: Radius.circular(8),
              bottomLeft: Radius.circular(8),
            ),
          ),
        ),
        child: Text('-'),
      ),
      Container(
        padding: EdgeInsets.symmetric(horizontal: 12, vertical: 7),
        color: Colors.grey[200],
        child: Text('${product.quantity} ?? 0'),
      ),
      GestureDetector(
        onTap: () => con.addItem(product),
        child: Container(
          padding: EdgeInsets.symmetric(horizontal: 12, vertical: 7),
          decoration: BoxDecoration(
            color: Colors.grey[200],
            borderRadius: BorderRadius.only(
              topRight: Radius.circular(8),

```

```

        bottomRight: Radius.circular(8),
      ),
    ),
    child: Text('+'),
  ),
),
],
);
}

Widget _imageProduct(Product product) {
  return Container(
    height: 70,
    width: 70,
    // padding: EdgeInsets.all(2),
    child: ClipRect(
      borderRadius: BorderRadius.circular(12),
      child: FadeInImage(
        image: product.imageUrl != null
          ? NetworkImage(product.imageUrl)
          : AssetImage('assets/img/no-image.png') as ImageProvider,
        fit: BoxFit.cover,
        fadeInDuration: Duration(milliseconds: 50),
        placeholder: AssetImage('assets/img/no-image.png'),
      ),
    ),
  );
}
}

```

Controlador de Crear Orden

```

import 'package:get/get.dart';
import 'package:get_storage/get_storage.dart';
import 'package:udemy_flutter_delivery/src/models/product.dart';
import 'package:udemy_flutter_delivery/src/pages/client/products/list/client_products_list_controller.dart';

class ClientOrdersCreateController extends GetxController {
  RxList<Product> selectedProducts = <Product>[].obs;
  var total = 0.0.obs;
  ClientProductsListController productsListController = Get.find();

  ClientOrdersCreateController() {
    if (GetStorage().read('shopping_bag') != null) {
      if (GetStorage().read('shopping_bag') is List<Product>) {
        var result = GetStorage().read('shopping_bag');
        selectedProducts.clear();
        selectedProducts.addAll(result);
      }
      else {
        var result = Product.fromJsonList(GetStorage().read('shopping_bag'));

```

```

        selectedProducts.clear();
        selectedProducts.addAll(result);
    }

    getTotal();

}

}

void getTotal() {
    total.value = 0.0;
    selectedProducts.forEach((product) {
        total.value = total.value + (product.quantity! * product.price!);
    });
}

void deleteItem(Product product) {
    selectedProducts.remove(product);
    GetStorage().write('shopping_bag', selectedProducts);
    getTotal();

    productsListController.items.value = 0;
    if (selectedProducts.length == 0) {
        productsListController.items.value = 0;
    }
    else {
        selectedProducts.forEach((p) {
            productsListController.items.value = productsListController.items.value + p.quantity!;
        });
    }
}

void addItem(Product product) {
    int index = selectedProducts.indexWhere((p) => p.id == product.id);
    selectedProducts.remove(product);
    product.quantity = product.quantity! + 1;
    selectedProducts.insert(index, product);
    GetStorage().write('shopping_bag', selectedProducts);
    getTotal();
    productsListController.items.value = 0;
    selectedProducts.forEach((p) {
        productsListController.items.value = productsListController.items.value + p.quantity!;
    });
}

void removeItem(Product product) {
    if (product.quantity! > 1) {
        int index = selectedProducts.indexWhere((p) => p.id == product.id);
        selectedProducts.remove(product);
        product.quantity = product.quantity! - 1;
        selectedProducts.insert(index, product);
        GetStorage().write('shopping_bag', selectedProducts);
        getTotal();
        productsListController.items.value = 0;
        selectedProducts.forEach((p) {
            productsListController.items.value = productsListController.items.value + p.quantity!;
        });
    }
}

```

```

}

void goToAddressList() {
  Get.toNamed('/client/address/list');
}
}

```

Órdenes del cliente en el mapa

```

import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:google_maps_flutter/google_maps_flutter.dart';
import 'package:udemy_flutter_delivery/src/pages/client/address/map/client_address_map_controller.dart';
import 'package:udemy_flutter_delivery/src/pages/client/orders/map/client_orders_map_controller.dart';
import 'package:udemy_flutter_delivery/src/pages/delivery/orders/map/delivery_orders_map_controller.dart';

class ClientOrdersMapPage extends StatelessWidget {
  ClientOrdersMapController controller = Get.put(ClientOrdersMapController());

  @override
  Widget build(BuildContext context) {
    return GetBuilder<ClientOrdersMapController>(
      builder: (value) => Scaffold(
        backgroundColor: Colors.grey[900],
        body: Stack(
          children: [
            Container(
              height: MediaQuery.of(context).size.height * 0.67,
              child: _googleMaps(),
            ),
            SafeArea(
              child: Column(
                children: [
                  Row(
                    mainAxisAlignment: MainAxisAlignment.spaceBetween,
                    children: [
                      _buttonBack(),
                      _iconCenterMyLocation(),
                    ],
                  ),
                  Spacer(),
                  _cardOrderInfo(context),
                ],
              ),
            ),
            // _buttonAccept(context)
          ],
        ),
      ));
  }

  Widget _buttonBack() {

```

```

return Container(
  alignment: Alignment.centerLeft,
  margin: EdgeInsets.only(left: 20),
  child: IconButton(
    onPressed: () => Get.back(),
    icon: Icon(
      Icons.arrow_back_ios,
      color: Colors.white,
      size: 30,
    ),
  ),
);
}

Widget _cardOrderInfo(BuildContext context) {
  return Container(
    height: MediaQuery.of(context).size.height * 0.33,
    width: double.infinity,
    decoration: BoxDecoration(
      color: Colors.grey[900],
      borderRadius: BorderRadius.only(
        topRight: Radius.circular(20),
        topLeft: Radius.circular(20),
      ),
    ),
    boxShadow: [
      BoxShadow(
        color: Colors.grey.withOpacity(0.3),
        spreadRadius: 4,
        blurRadius: 6,
        offset: Offset(0, 3)
      )
    ],
  ),
  child: Column(
    children: [
      _listTileAddress(
        con.order.address?.neighborhood ?? "",
        'Barrio',
        Icons.my_location
      ),
      _listTileAddress(
        con.order.address?.address ?? "",
        'Direccion',
        Icons.location_on
      ),
      Divider(color: Colors.grey, endIndent: 30, indent: 30),
      _deliveryInfo(),
    ],
  ),
);
}

Widget _deliveryInfo() {
  return Container(
    margin: EdgeInsets.symmetric(horizontal: 35, vertical: 20),
    child: Row(
      children: [
        _imageClient(),

```

```

        SizedBox(width: 15),
        Text(
          '${con.order.delivery?.name ?? ""} ${con.order.delivery?.lastname ?? ""}',
          style: TextStyle(
            color: Colors.white,
            fontSize: 16,
            fontWeight: FontWeight.bold
          ),
          maxLines: 1,
        ),
        Spacer(),
        Container(
          decoration: BoxDecoration(
            borderRadius: BorderRadius.all(Radius.circular(15)),
            color: Colors.grey[200]
          ),
          child: IconButton(
            onPressed: () => con.callNumber(),
            icon: Icon(Icons.phone, color: Colors.black),
          ),
        ),
      ],
    ),
  );
}

Widget _imageClient() {
  return Container(
    height: 50,
    width: 50,
    // padding: EdgeInsets.all(2),
    child: ClipRRect(
      borderRadius: BorderRadius.circular(12),
      child: FadeInImage(
        image: con.order.delivery!.image != null
          ? NetworkImage(con.order.delivery!.image!)
          : AssetImage('assets/img/no-image.png') as ImageProvider,
        fit: BoxFit.cover,
        fadeInDuration: Duration(milliseconds: 50),
        placeholder: AssetImage('assets/img/no-image.png'),
      ),
    ),
  );
}

Widget _listTileAddress(String title, String subtitle, IconData iconData) {
  return Container(
    margin: EdgeInsets.symmetric(horizontal: 20),
    child: ListTile(
      title: Text(
        title,
        style: TextStyle(
          fontSize: 13,
          color: Colors.white
        ),
      ),
      subtitle: Text(
        subtitle,

```

```

        style:
            color:
                TextStyle(
                    Colors.white
                ),
            ),
        trailing:
            Icon(iconData,
                color:
                    Colors.white,
            ),
        );
    }

Widget _iconCenterMyLocation() {
    return
        onTap:
            () =>
        child:
            alignment:
                Alignment.centerRight,
            margin:
                EdgeInsets.symmetric(horizontal:
                    5),
            child:
                shape:
                    CircleBorder(),
                color:
                    Colors.white,
                elevation:
                    4,
            child:
                padding:
                    EdgeInsets.all(10),
                child:
                    Icons.location_searching,
                color:
                    Colors.grey[600],
                size:
                    20,
            ),
        ),
    ),
);
}

Widget _googleMaps() {
    return
        initialCameraPosition:
            con.initialPosition,
        mapType:
            MapType.normal,
        onMapCreated:
            con.onMapCreate,
        myLocationButtonEnabled:
            false,
        myLocationEnabled:
            false,
        markers:
            Set<Marker>.of(con.markers.values),
        polylines:
            con.polylines,
    );
}
}

```

Controlador

```

import
    'dart:async';

import
    'package:flutter/material.dart';
import
    'package:flutter_phone_direct_caller/flutter_phone_direct_caller.dart';
import
    'package:flutter_polyline_points/flutter_polyline_points.dart';
import
    'package:fluttertoast/fluttertoast.dart';
import
    'package:geocoding/geocoding.dart';
import
    'package:geolocator/geolocator.dart';
import
    'package:get/get.dart';

```

```

import 'package:google_maps_flutter/google_maps_flutter.dart';
import 'package:location/location.dart' as location;
import 'package:socket_io_client/socket_io_client.dart';
import 'package:udemy_flutter_delivery/src/environment/environment.dart';
import 'package:udemy_flutter_delivery/src/models/order.dart';
import 'package:udemy_flutter_delivery/src/providers/orders_provider.dart';

class ClientOrdersMapController extends GetxController {
  Socket = io('${Environment.API_URL}orders/delivery', <String, dynamic> {
    'transports': ['websocket'],
    'autoConnect': false
  });

  Order = Order.fromJson(Get.arguments['order'] ?? {});
  OrdersProvider = OrdersProvider();

  CameraPosition initialPosition = CameraPosition(
    target: LatLng(1.2004567, -77.2787444),
    zoom: 14
  );

  LatLng? addressLatLng;
  var addressName = ".obs";

  Completer<GoogleMapController> mapController = Completer();
  Position? position;

  Map<MarkerId, Marker> markers = <MarkerId, Marker>{ }.obs;
  BitmapDescriptor? deliveryMarker;
  BitmapDescriptor? homeMarker;

  StreamSubscription? positionSubscribe;

  Set<Polyline> polylines = <Polyline>{ }.obs;
  List<LatLng> points = [];

  ClientOrdersMapController() {
    print('Order: ${order.toJson()}');
  }

  checkGPS(); // VERIFICAR SI EL GPS ESTA ACTIVO
  connectAndListen();
}

void connectAndListen() {
  socket.connect();
  socket.onConnect((data) {
    print('ESTE DISPISITIVO SE CONECTO A SOCKET IO');
  });
  listenPosition();
  listenToDelivered();
}

void listenPosition() {
  socket.on('position/${order.id}', (data) {
    addMarker(
      'delivery',

```



```

        data['lat'],
        data['lng'],
        'Tu',
        "
    },
    deliveryMarker!
  );
});
}

void listenToDelivered() {
  socket.on('delivered/${order.id}', (data) {
    Fluttertoast.showToast(
      msg: 'El estado de la orden se actualizo a entregado',
      toastLength: Toast.LENGTH_LONG
    );
    Get.offNamedUntil('/client/home', (route) => false);
  });
}

Future setLocationDraggableInfo() async {
  double lat = initialPosition.target.latitude;
  double lng = initialPosition.target.longitude;

  List<Placemark> address = await placemarkFromCoordinates(lat, lng);

  if (address.isNotEmpty) {
    String direction = address[0].thoroughfare ?? "";
    String street = address[0].subThoroughfare ?? "";
    String city = address[0].locality ?? "";
    String department = address[0].administrativeArea ?? "";
    String country = address[0].country ?? "";
    addressName.value = '$direction # $street, $city, $department';
    addressLatLng = LatLng(lat, lng);
    print('LAT Y LNG: ${addressLatLng?.latitude ?? 0} ${addressLatLng?.longitude ?? 0}');
  }
}

void selectRefPoint(BuildContext context) {
  if (addressLatLng != null) {
    Map<String, dynamic> data = {
      'address': addressName.value,
      'lat': addressLatLng!.latitude,
      'lng': addressLatLng!.longitude,
    };
    Navigator.pop(context, data);
  }
}

Future<BitmapDescriptor> createMarkerFromAssets(String path) async {
  ImageConfiguration configuration = ImageConfiguration();
  BitmapDescriptor descriptor = await BitmapDescriptor.fromAssetImage(
    configuration,
    path
  );
}

```

```

return descriptor;
}

void addMarker(
String markerId,
double lat,
double lng,
String title,
String content,
BitmapDescriptor iconMarker
) {
MarkerId id = MarkerId(markerId);
Marker marker = Marker(
markerId: id,
icon: iconMarker,
position: LatLng(lat, lng),
infoWindow: InfoWindow(title: title, snippet: content)
);

markers[id] = marker;

update();
}

void checkGPS() async {
deliveryMarker = await createMarkerFromAssets('assets/img/delivery_little.png');
homeMarker = await createMarkerFromAssets('assets/img/home.png');

bool isLocationEnabled = await Geolocator.isLocationServiceEnabled();

if (isLocationEnabled == true) {
updateLocation();
}
else {
bool locationGPS = await location.Location().requestService();
if (locationGPS == true) {
updateLocation();
}
}
}

Future<void> setPolylines(LatLng from, LatLng to) async {
PointLatLng pointFrom = PointLatLng(from.latitude, from.longitude);
PointLatLng pointTo = PointLatLng(to.latitude, to.longitude);
PolylineResult result = await PolylinePoints().getRouteBetweenCoordinates(
Environment.API_KEY_MAPS,
pointFrom,
pointTo
);

for (PointLatLng point in result.points) {
points.add(LatLng(point.latitude, point.longitude));
}

Polyline polyline = Polyline(
polylineId: PolylineId('poly'),
color: Colors.amber,
points: points,
);
}

```

```

        width: 5
    );

    polylines.add(polyline);
    update();
}

void updateLocation() async {
  try{
    await _determinePosition();
    position = await Geolocator.getLastKnownPosition(); // LAT Y LNG (ACTUAL)
    animateCameraPosition(order.lat ?? 1.2004567, order.lng ?? -77.2787444);

    addMarker(
      'delivery',
      order.lat ?? 1.2004567,
      order.lng ?? -77.2787444,
      "Tu repartidor",
      deliveryMarker!
    );

    addMarker(
      'home',
      order.address?.lat ?? 1.2004567,
      order.address?.lng ?? -77.2787444,
      "Lugar de entrega",
      homeMarker!
    );

    LatLng from = LatLng(order.lat ?? 1.2004567, order.lng ?? -77.2787444);
    LatLng to = LatLng(order.address?.lat ?? 1.2004567, order.address?.lng ?? -77.2787444);

    setPolylines(from, to);

  } catch(e) {
    print('Error: ${e}');
  }
}

void callNumber() async{
  String number = order.delivery?.phone ?? ""; //set the number here
  await FlutterPhoneDirectCaller.callNumber(number);
}

void centerPosition() {
  if (position != null) {
    animateCameraPosition(position!.latitude, position!.longitude);
  }
}

Future animateCameraPosition(double lat, double lng) async {
  GoogleMapController controller = await mapController.future;
  controller.animateCamera(CameraUpdate.newCameraPosition(
    CameraPosition(
      target: LatLng(lat, lng),

```

```

        zoom: 13,
        bearing: 0
    )
    ));
}

Future<Position> _determinePosition() async {
  bool serviceEnabled;
  LocationPermission permission;

  serviceEnabled = await Geolocator.isLocationServiceEnabled();
  if (!serviceEnabled) {
    return Future.error('Location services are disabled.');
```

```

  }

  permission = await Geolocator.checkPermission();
  if (permission == LocationPermission.denied) {
    permission = await Geolocator.requestPermission();
    if (permission == LocationPermission.denied) {
      return Future.error('Location permissions are denied.');
```

```

    }
  }

  if (permission == LocationPermission.deniedForever) {
    return Future.error(
      'Location permissions are permanently denied, we cannot request permissions.');
```

```

  }

  return await Geolocator.getCurrentPosition();
}

void onMapCreate(GoogleMapController controller) {
  controller.setMapStyle([{"elementType":"geometry","stylers":[{"color":"#212121"}]},{"elementType":"labels.icon","stylers":[{"visibility":"off"}]},{"elementType":"labels.text.fill","stylers":[{"color":"#757575"}]},{"elementType":"labels.text.stroke","stylers":[{"color":"#212121"}]},{"featureType":"administrative","elementType":"geometry","stylers":[{"color":"#757575"}]},{"featureType":"administrative.country","elementType":"labels.text.fill","stylers":[{"color":"#9e9e9e"}]},{"featureType":"administrative.land_parcel","stylers":[{"visibility":"off"}]},{"featureType":"administrative.locality","elementType":"labels.text.fill","stylers":[{"color":"#bdbdbd"}]},{"featureType":"poi","elementType":"labels.text.fill","stylers":[{"color":"#757575"}]},{"featureType":"poi.park","elementType":"geometry","stylers":[{"color":"#181818"}]},{"featureType":"poi.park","elementType":"labels.text.fill","stylers":[{"color":"#616161"}]},{"featureType":"poi.park","elementType":"labels.text.stroke","stylers":[{"color":"#1b1b1b"}]},{"featureType":"road","elementType":"geometry.fill","stylers":[{"color":"#2c2c2c"}]},{"featureType":"road","elementType":"labels.text.fill","stylers":[{"color":"#8a8a8a"}]},{"featureType":"road.arterial","elementType":"geometry","stylers":[{"color":"#373737"}]},{"featureType":"road.highway","elementType":"geometry","stylers":[{"color":"#3c3c3c"}]},{"featureType":"road.highway.controlled_access","elementType":"geometry","stylers":[{"color":"#4e4e4e"}]},{"featureType":"road.local","elementType":"labels.text.fill","stylers":[{"color":"#616161"}]},{"featureType":"transit","elementType":"labels.text.fill","stylers":[{"color":"#757575"}]},{"featureType":"water","elementType":"geometry","stylers":[{"color":"#000000"}]},{"featureType":"water","elementType":"labels.text.fill","stylers":[{"color":"#3d3d3d"}]
  ]});
  mapController.complete(controller);
}

```

```
@override
void onClose() {
  // TODO: implement onClose
  super.onClose();
  socket.disconnect();
  positionSubscribe?.cancel();
}
```

4.4. Pruebas

4.4.1. Análisis de requisitos

El análisis de requisitos realizado a la aplicación móvil fue de vital importancia en el desarrollo de la aplicación, lo cual nos ayudó a comprender las necesidades, expectativas y restricciones, estableciendo una base sólida para la comunicación y colaboración entre las partes interesadas.

Análisis de los requisitos de la aplicación móvil:

Requisitos del propietario del local:

Registro de Usuario

El dueño del local puede iniciar sesión en la aplicación utilizando el correo y contraseña, con el que fue registrado caso contrario la aplicación no le permitirá ingresar. Una vez iniciado sesión, el usuario puede acceder a las funcionalidades de la aplicación.

Descripción del producto

El dueño del local puede agregar productos a la aplicación, en el cual tiene un apartado para agregarle una descripción, la cual se guarda en el sistema y estará disponible para que los clientes la vean.

Publicación del Menú

El usuario debe ingresar los platos y bebidas que oferta que están disponibles en su negocio, de esta manera el menú estará actualizado y disponible para que los clientes lo vean.

Notificaciones

El usuario puede enviar notificaciones a los clientes sobre el estado del pedido, en otras palabras, si está en preparación, en camino o a su vez despachado.

Requisitos del Cliente

Registro de Usuario

El cliente puede registrarse en el cual se le pide ingresar sus datos personales tales como la cédula, nombre, apellido, email, teléfono y contraseña, una vez ingresada esta información y si los datos son correctos entonces se creará el usuario para que pueda iniciar sesión correctamente y pueda acceder a los servicios de la aplicación.

Método de pago

El cliente puede seleccionar el método de pago disponible en el local el cual contara con transferencia o pago en efectivo, según su preferencia.

Realizar pedido

El cliente puede seleccionar los platos y bebidas de su preferencia y confirmar el pedido, el cual llega al usuario propietario para ser visualizado, preparado y despachado.

Búsqueda de productos

El cliente puede realizar búsquedas de los platos y bebidas en la aplicación a través de una barra de búsqueda la cual le permite encontrar de forma rápida los alimentos que desea visualizar.

Geolocalización

Para acceder a esta funcionalidad el cliente debe activar el GPS de su dispositivo, otorgándole los permisos a la aplicación para acceder a su ubicación, la cual le permite obtener la ubicación actual del repartidor, que se podrá visualizar en un mapa.

4.4.2. Planificación de pruebas

N°	CRONOGRAMA DE ACTIVIDADES DE PRUEBAS	
		3 SEMANAS

	TIEMPO ESTIMADO	4 días	5 días	3 días	6 días	3 días
1	Análisis de requisitos					
2	Diseño y desarrollo de casos de prueba					
3	Configuración del entorno de prueba					
4	Ejecución de la prueba					
5	Prueba de cierre					

Ganchala L, Ocaña H. *Planificación de Pruebas.*; 2023.

4.4.3. Diseño y desarrollo de casos de prueba

El objetivo de las pruebas realizadas a la aplicación móvil “Comedor UEB”, es asegurar la calidad, permitiendo identificar errores que puedan alterar el funcionamiento, de esta manera nos ayuda a mejorar la experiencia de los usuarios garantizando que la aplicación cumpla con las funcionalidades de manera eficiente y confiable.

Las pruebas que se llevaron a cabo son las siguientes:

Usabilidad

Funcionalidad

Rendimiento

Seguridad

Disponibilidad

Para el desarrollo de las pruebas se diseñó con el siguiente formato:

Nº

Objetivo

Precondición

Descripción de la prueba

Resultados esperados

Resultados obtenidos

4.4.4. Configuración del entorno de prueba

Para realizar las pruebas a la aplicación móvil necesitamos:

Hardware

Teléfono con Android y conexión a internet

Software

Herramienta sonarQube, FireBase test de robo.

Equipo de Testing

Responsables

Henry Ocaña

Lenon Ganchala

Actividad

Realizar las diferentes pruebas a la aplicación móvil

4.4.5. Ejecución de la prueba

Pruebas de Usabilidad

Para el desarrollo de las pruebas de usabilidad de la aplicación móvil se consideró el uso y la experiencia del usuario.

Cliente

N°	1
Objetivo	Iniciar sesión en la aplicación móvil
Precondición	El usuario debe contar con un celular Android, con la aplicación instalada y conectado a internet.
Descripción de la prueba	El usuario llenará los campos de correo y contraseña, para luego presionar el botón de login
Resultados esperados	El usuario logró iniciar sesión
Resultados obtenidos	Se logró iniciar sesión correctamente

N°	2
Objetivo	Registrar un nuevo usuario
Precondición	El usuario debe contar con un correo electrónico
Descripción de la prueba	El usuario debe seleccionar “regístrate aquí” y pasar a la pantalla de registro y llenar los campos (seleccionar imagen, correo, nombre, apellido, cédula, teléfono, contraseña y confirmar contraseña) y presionar registrarse
Resultados esperados	El usuario ingresa a la pantalla de registro y registra el nuevo usuario
Resultados obtenidos	Se creó el nuevo usuario satisfactoriamente.
N°	3
Objetivo	Visualizar las categorías y los productos disponibles
Precondición	El usuario debe haber iniciado sesión en la aplicación
Descripción de la prueba	El usuario debe revisar las categorías disponibles desplazándose por cada una de ellas visualizando los productos de cada una
Resultados esperados	El usuario puede desplazarse por cada una de las categorías y visualizar los productos
Resultados obtenidos	Se logró seleccionar cada categoría y visualizar los productos con éxito

N°	4
Objetivo	Seleccionar los productos modificando la cantidad desea para confirmar el pedido
Precondición	Haber visualizado los productos y las categorías
Descripción de la prueba	El usuario debe seleccionar un producto, modificar la cantidad deseada y confirmar el pedido
Resultados esperados	El usuario selecciono el producto y confirmo el pedido
Resultados obtenidos	Se seleccionó el producto y se confirmó el pedido satisfactoriamente.

N°	5
Objetivo	Buscar productos a través de la barra de búsqueda y visualizar los pedidos
Precondición	El usuario debe estar en la sección de productos
Descripción de la prueba	El usuario debe pulsar en la barra de búsqueda e ingresar el nombre de un producto y después pasar a la sección de mis pedidos
Resultados esperados	El usuario realizo la búsqueda y encontró el producto, después se desplazó a la sección de mis pedidos y visualizo las ordenes confirmadas
Resultados obtenidos	Se logró buscar el producto y desplazarse a la sección de mis pedidos satisfactoriamente

Propietario del local

N°	1
Objetivo	Iniciar sesión en la aplicación y seleccionar el rol restaurante
Precondición	Debe utilizar el correo y contraseña de administrador
Descripción de la prueba	El usuario debe ingresar el correo y la contraseña, presionar login y seleccionar el rol de restaurante
Resultados esperados	El usuario lleno los campos con los datos correctos, e ingreso a la aplicación y selecciono el rol restaurante
Resultados obtenidos	Se ingresó a la aplicación y se seleccionó el rol restaurante sin dificultad
N°	3
Objetivo	Seleccionar la sección categoría y registrar una nueva categoría
Precondición	Crear una nueva categoría
Descripción de la prueba	Debe seleccionar la sección categoría e ingresar el nombre, una descripción y presionar crear categoría
Resultados esperados	El usuario selecciono el apartado de categoría ingreso la información y crea la categoría
Resultados obtenidos	Se creó la categoría satisfactoriamente

N°	4
Objetivo	Crear un producto nuevo
Precondición	Seleccionar el apartado de nuevo producto
Descripción de la prueba	Ingresar a la sección de nuevo producto e ingresar la información (nombre, descripción, precio, seleccionar la categoría y seleccionar tres imágenes del producto), y presionar el botón crear producto.
Resultados esperados	El usuario ingresa la información selecciona la categoría, carga las tres imágenes y crea el producto
Resultados obtenidos	Se creó el producto exitosamente
N°	5
Objetivo	Visualizar el pedido y despachar
Precondición	Ingresar al apartado de Pedidos
Descripción de la prueba	El usuario debe pulsar en pedidos y visualizar las órdenes, para pasarlas al estado despachado.
Resultados esperados	El usuario puede visualizar las órdenes confirmadas y cambiar la orden ha despachado
Resultados obtenidos	Se visualizó las órdenes a preparar y se les asignó el estado de despachado sin dificultad

Repartidor

N°	1
Objetivo	Visualizar pedidos y pasar al estado de entregado
Precondición	Ingresar al rol de repartidor
Descripción de la prueba	Entrar a la sección de pedidos para visualizar las órdenes a entregar, para pasar al estado entregado
Resultados esperados	El usuario visualiza los pedidos y los pasa al estado entregado
Resultados obtenidos	Se visualizó los pedidos a entregar y se cambió al estado entregado satisfactoriamente.

Pruebas de Funcionalidad

Las pruebas de funcionalidad se encargan de visualizar si la aplicación está ejecutando correctamente las pantallas dentro de la aplicación móvil

Pruebas de Funcionalidad			
Objetivo	Verificar que el sistema apruebe todos los requerimientos de parte del cliente y que todas sus paginas funcionen correctamente		
Fecha	19/10/2023		
N°	Criterios	Cumple	
		SI	NO
1	Iniciar sesión en la aplicación móvil	X	
2	Registrar un nuevo usuario	X	
3	Visualizar las categorías y los productos disponibles	X	
4	Seleccionar los productos modificando la cantidad desea para confirmar el pedido	X	
5	Buscar productos a través de la barra de busque y visualizar los pedidos	X	
7	Iniciar sesión en la aplicación y seleccionar el rol restaurante	X	
8	Seleccionar la sección categoría y registrar una nueva categoría	X	
9	Crear un producto nuevo	X	
10	Visualizar el pedido y despachar	X	
11	Visualizar pedidos y pasar al estado de entregado	X	

Ganchala L, Ocaña H. *Pruebas de Funcionalidad.*; 2023.

Pruebas de Rendimiento

Las pruebas de rendimiento se encargan de estresar a la aplicación para ver posibles fallos que se puedan dar y corregirlos a tiempo para su despliegue

Pruebas de Rendimiento			
Objetivo	Evaluar el rendimiento de la aplicación móvil		
Fecha	19/10/2023		
N°	Criterios	Cumple	
		SI	NO
1	La aplicación indica rapidez al subir y visualizar el contenido como texto o imágenes.	X	
2	La aplicación expone una velocidad normal en el agregado visible de los contenedores.	X	
3	La aplicación expone un tiempo de subida rápido para el contenido que ocupa toda la parte principal (por ejemplo, los elementos de roles y su texto).	X	
4	La aplicación móvil se ajusta a varios dispositivos para su correcta visualización en distintas gamas de celulares.	X	
5	La aplicación móvil representa un diseño acorde durante el proceso de carga de la aplicación.	X	

Ganchala L, Ocaña H. *Pruebas de Rendimiento.*; 2023.

Pruebas de Seguridad

Las pruebas de seguridad se encargan de evaluar si el sistema funciona adecuadamente ante ataques y ver si existen vulnerabilidades en el código

Pruebas de Seguridad			
Objetivo	Evaluar la seguridad de la aplicación móvil		
Fecha	19/10/2023		
N°	Criterios	Cumple	
		SI	NO
1	Control de Inicio de sesión	X	
2	Criptografía Registrada en la base de datos	X	
3	Prevención de inyecciones	X	
4	Diseño Seguro	X	
5	Componentes Actualizados	X	
7	Autenticación Segura	X	
8	Validación por Google Fire Base	X	
9	Integridad De los Datos	X	
10	Protección contra solicitudes en las validaciones	X	

Ganchala L, Ocaña H. *Pruebas de Seguridad.*; 2023.

Pruebas de Disponibilidad

Pruebas de Disponibilidad se encargan de que la aplicación móvil funcione correctamente el tiempo de vida de la aplicación

Pruebas de Disponibilidad			
Objetivo	Evaluar la disponibilidad de la aplicación móvil		
Fecha	19/10/2023		
N°	Criterios	Cumple	
		SI	NO
1	La aplicación funciona las 24/7, con la restricción en caso de los periodos que se le de mantenimiento a la aplicación	X	
2	Existen planes de contingencia para la aplicación móvil		x
3	Capacidad de recuperación en caso de errores	X	
4	Capacidad operativa en caso de que no carguen imágenes	X	

Ganchala L, Ocaña H. *Pruebas de Disponibilidad.*; 2023.

4.4.6. Prueba de cierre

Prueba de Cierre				
Objetivo		Resultado Obtenido de las pruebas		
Fecha		19/10/2023		
N°	1=Deficiente 2=Aceptable 3=Excelente			
	Pruebas	Metrica interna	Valoracion promedio	Porcentaje de valoracion
1	Funcionalidad	Verificar que el sistema apruebe todos los requerimientos de parte del cliente y que todas sus paginas funcionen correctamente	3	100%
2	Seguridad	Evaluar la seguridad de la aplicacion movil	3	100%
3	Rendimiento	Evaluar el rendimiento de la aplicacion movil	3	100%
4	Disponibilidad	Evaluar la disponibilidad de la aplicacion movil	2,25	75%
5	Usabilidad	Evaluar el grado de conformidad con la app	2,25	75%
TOTAL			2,7	90%
Observaciones: El nivel de aceptacion de la aplicacion movil tiene un nivel de calidad apto para la distribucion de la aplicacion sin embargo se podria mejorar en futuras actualizaciones de la aplicacion				

Ganchala L, Ocaña H. *Pruebas de cierre.*; 2023.

CONCLUSIONES

Los locales del comedor universitario de la Universidad Estatal de Bolívar tienen una gestión ineficiente de pedidos de alimentos realizados por sus clientes.

La aplicación móvil “Comedor UEB” utiliza la Ingeniería de Software, y herramientas informáticas que permiten solucionar la problemática, con soluciones tecnológicas innovadoras.

Las pruebas realizadas a la aplicación móvil permiten demostrar su calidad, para generar confianza y libertad en los usuarios en la solicitud de pedidos, difusión del menú, con lo cual mejora la gestión e información sobre los servicios de alimentación que ofrecen.

RECOMENDACIONES

Establecer un sistema de retroalimentación constante con los usuarios y propietarios de los locales para recopilar comentarios y sugerencias que permitan la evolución de la aplicación.

Considerar la expansión de la aplicación a otras plataformas móviles, como iOS, para abarcar más usuarios y que la acogida de la aplicación en la comunidad universitaria sea más amplia.

Realizar mantenimientos a la aplicación móvil para adaptarse a las nuevas y futuras tendencias tecnológicas, fomentando un mayor uso a largo plazo.

BIBLIOGRAFÍA

- Álvarez Córdova, B. S., & Falconí Flor, L. A. (2021). Desarrollo de sistema web y aplicación móvil informativa de restaurante de alrededores a la escuela Politécnica Nacional. Recuperado de <http://bibdigital.epn.edu.ec/handle/15000/21415>
- API: Mulesoft. (2021). What is an API? Recuperado de <https://www.mulesoft.com/resources/api/what-is-an-api>
- Aplicación informática: TechTerms. (2021). Application Definition. Recuperado de <https://techterms.com/definition/application>
- Authentication y authorization: Microsoft. (2021). Authentication and authorization in Azure App Service. Recuperado de <https://docs.microsoft.com/en-us/azure/app-service/overview-authentication-authorization>
- Base de datos: Oracle. (2021). What is a Database? Recuperado de <https://www.oracle.com/database/what-is-database/>
- Cabrera García, C. A., & Allain Manrique, Y. J. (2020). Diseño y desarrollo de una aplicación informática para la gestión eficiente de comandas del restaurante Gaitano resto bar del Autónoma de Ica, 2020 [Universidad Ica]. Recuperado de <http://repositorio.autonomadeica.edu.pe/handle/autonomadeica/1122>
- Cádiz, D. (2016, 22 de septiembre). La revolución de las aplicaciones para móviles. Diario de Cádiz. Recuperado de https://www.diariodecadiz.es/cadiz/revolucion-aplicaciones-moviles_0_1064593684.html
- Ciencias. (s.f.). Metodología para el diseño y desarrollo de aplicaciones móviles. Recuperado de: <https://www.3ciencias.com/libros/libro/metodologia-para-el-diseno-y-desarrollo-de-aplicaciones-moviles/>
- Experiencia del usuario (UX): Nielsen Norman Group. (2021). The Definition of User Experience (UX). Recuperado de <https://www.nngroup.com/articles/definition-user-experience/>
- Flores Torrez, A. (2021). Aplicación móvil de e-commerce para la gestión de pedidos a proveedores de empresas gastronómicas (Doctoral dissertation).
- Geolocalización: Techopedia. (2021). Geolocation. Recuperado de <https://www.techopedia.com/definition/713/geolocation>

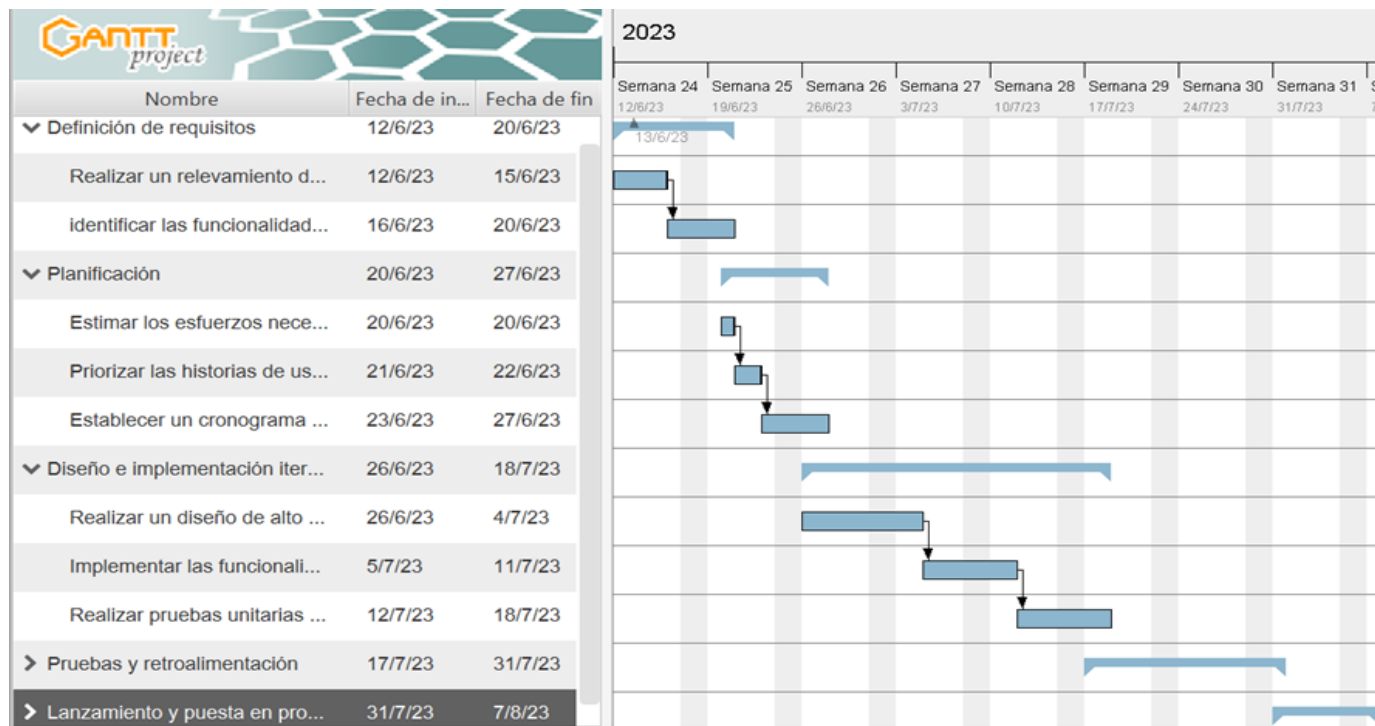
- Gestión de inventario y menús: Toast. (2021). Restaurant Inventory Management. Recuperado de <https://pos.toasttab.com/resources/restaurant-inventory-management>
- Interfaz de usuario (UI): Usability.gov. (2021). User Interface Design Basics. Recuperado de <https://www.usability.gov/what-and-why/user-interface-design.html>
- Mocha Zhingri, C. R., & Ochoa Calderón, J. E. (2022). Diseño y desarrollo de un sistema web y aplicación móvil mediante angular e ionic para la gestión de pedidos en restaurantes (Bachelor's thesis).
- Modelo Vista Controlador (MVC): Microsoft. (2021). Overview of ASP.NET Core MVC. Recuperado de <https://docs.microsoft.com/en-us/aspnet/core/mvc/overview?view=aspnetcore-6.0>
- Notificaciones push: OneSignal. (2021). What Are Push Notifications? Recuperado de <https://onesignal.com/blog/what-are-push-notifications/>
- Pasarelas de pagos: PayPal. (2021). What is a payment gateway and how does it work? Recuperado de <https://www.paypal.com/us/brc/article/payment-gateway>
- Sistema de pedidos: Shopify. (2021). Order Management System. Recuperado de <https://www.shopify.com/order-management-system>
- Sistema de seguimiento de pedidos: FedEx. (2021). Tracking Your Shipment or Packages. Recuperado de <https://www.fedex.com/en-us/tracking.html>
- Sommerville, I., & Velázquez, S. F. (2011). Ingeniería de software.
- Tubón Cando, G. A. (2020). Aplicación móvil con Georreferenciación para gestión de pedidos a domicilio de un local de comida (Bachelor's thesis, Pontificia Universidad Católica del Ecuador).

ANEXOS 1

Cronograma (Gantt)

Cronograma

Las actividades del trabajo de integración tecnológico fueron desarrolladas acorde a las fechas establecidas.



Ganchala L, Ocaña H. *Cronograma de actividades.*; 2023.

ANEXOS 2

Presupuesto Ejecutado

Configuración de la Base de Datos (BD):

- Instalar y configurar BD: Se estima que esta tarea puede llevar de 4 horas (en el caso más simple) hasta 40 horas (en el caso más difícil). La instalación y configuración de una base de datos es una parte fundamental de la infraestructura de la aplicación. La estimación se basa en la complejidad de la configuración.
- Crear entidades de BD: Esta tarea se estima en 16 horas. La creación de entidades en la base de datos probablemente involucra diseñar tablas y establecer relaciones, lo que puede requerir tiempo.

Creación de la Aplicación:

- Crear Servicios: Se estima en 32 horas. Esto implica desarrollar servicios que la aplicación utilizará para funcionar correctamente. La estimación no incluye la complejidad, por lo que podría variar dependiendo de la cantidad y complejidad de los servicios necesarios.
- Crear Relación con las API de Google: Se estima en 16 horas. Esta tarea involucra integrar la aplicación con las API de Google, lo que podría requerir tiempo para comprender y conectar adecuadamente con estas API.
- Crear el BackEnd en NodeJS: Se estima en 32 horas. Desarrollar el BackEnd de una aplicación puede ser un trabajo complejo que involucra la lógica de negocios y la gestión de datos.
- Crear el FrontEnd en Flutter: Se estima en 32 horas. El desarrollo del FrontEnd implica crear la interfaz de usuario y la experiencia del usuario de la aplicación. Flutter es un marco de desarrollo para aplicaciones móviles y web, por lo que esta estimación incluye el tiempo necesario para aprender y utilizar Flutter.
- Instalación y configuración del servidor remoto: Se estima en 24 horas. Esto implica configurar un servidor remoto para alojar la aplicación. La estimación no incluye detalles sobre el proveedor de servicios de alojamiento o la infraestructura.

Pruebas:

- Usabilidad: Se estima en 24 horas. La usabilidad se refiere a la facilidad de uso de la aplicación, por lo que esta tarea podría incluir pruebas de usuario y mejoras en la interfaz de usuario.

- **Mantenibilidad:** Se estima en 16 horas. La mantenibilidad se refiere a la capacidad de realizar futuras actualizaciones y correcciones en la aplicación. Esto podría implicar documentación y organización del código.
- **Rendimiento:** Se estima en 24 horas. Las pruebas de rendimiento evalúan la velocidad y eficiencia de la aplicación. Esto podría implicar la optimización del código.
- **Seguridad:** Se estima en 32 horas. La seguridad es fundamental en cualquier aplicación, y esta estimación podría incluir pruebas de seguridad y la implementación de medidas de protección.

Gran Total de Esfuerzo: 336 horas.

- El proyecto en su conjunto requerirá 336 horas de trabajo. Esto se traduce en 42 días laborables de 8 horas cada uno, o alrededor de 10.5 semanas de trabajo a tiempo completo. También se puede considerar como aproximadamente 2.1 meses de trabajo a tiempo completo.

Tabla de costos por módulos y componentes

Estimación de esfuerzos por módulos o componentes

Configuración del Base de Datos (BD)	Simple	Muy fácil	Fácil	Normal	Difícil	Muy Difícil	Horas
instalar y configurar BD			1				16
Crear entidades de BD			1				16
Crear Servicios					1		32
Total Esfuerzo							64

Creación De la Aplicación	Simple	Muy fácil	Fácil	Normal	Difícil	Muy Difícil	Horas
Crear Servicios					1		32
Crear Relación con las Api de Google			1				16
Crear El BackEnd en NodeJS					1		32
Crear El FrontEnd En Flutter					1		32
Instalación y configuración servidor remoto				1			24
Total Esfuerzo							136

Pruebas	Simple	Muy fácil	Fácil	Normal	Difícil	Muy Difícil	Horas
Usabilidad				1			24
Mantenibilidad			1				16
Rendimiento				1			24
Seguridad					1		32
Total Esfuerzo							96

Gran Total							336
-------------------	--	--	--	--	--	--	-----

	Simple	Muy fácil	Fácil	Normal	Difícil	Muy Difícil
Horas	4	8	16	24	32	40

Total de esfuerzos			
336	42	10,5	2,1

Agenda y Recursos				
Días	42	5,25	0,7	0
Semanas	10,5	1,05	0	0
Meses	2,1	0	0	0

Costo del proyecto				
Horas	336	168	112	84
Semanas	10,5	1,05	0	0

Meses	2,1	0	0	0
Total	\$ 5.040,00	\$ 5.040,00	\$ 5.040,00	\$ 5.040,00

Desglose de Proyecto				
Desarrollador (50%)	\$ 2.520,00	\$ 2.520,00	\$ 2.520,00	\$ 2.520,00
Costos Fijos (30%)	\$ 1.512,00	\$ 1.512,00	\$ 1.512,00	\$ 1.512,00
Gastos Extra (10%)	\$ 504,00	\$ 504,00	\$ 504,00	\$ 504,00
Ganancia (10%)	\$ 504,00	\$ 504,00	\$ 504,00	\$ 504,00
Total	\$ 5.040,00	\$ 5.040,00	\$ 5.040,00	\$ 5.040,00

Costo x Hora \$ 15,00

ANEXOS 3

Instrumentos de recopilación de datos



UNIVERSIDAD ESTATAL DE BOLÍVAR

FACULTAD DE CIENCIAS ADMINISTRATIVAS GESTIÓN EMPRESARIAL E INFORMÁTICA



ENTREVISTA DUEÑOS DE LOS LOCALES DEL COMEDOR

Objetivo: Recopilar información y requisitos para desarrollar la aplicación móvil Comedor UEB, que mejore la venta de comida para los clientes. La participación es agradecida y la información proporcionada será confidencial.

1. ¿Cuál es su experiencia en el uso de aplicaciones móviles de venta de comida?
2. ¿Qué tipo de comida o productos vende actualmente su negocio?
3. ¿Cuáles son los inconvenientes que enfrenta al ofertar comida?
4. ¿Qué información y detalles le gustaría mostrar sobre los productos en la aplicación? (por ejemplo, descripciones, ingredientes, precios, imágenes)
5. ¿Qué funcionalidades considera esenciales para una aplicación móvil de venta de comida? (por ejemplo, registro de usuario, publicación de menús, subir imágenes de los productos, notificaciones del estado del producto)
6. ¿Preferiría que la aplicación móvil tuviera una función de búsqueda para encontrar rápidamente los productos deseados?
7. ¿Qué métodos de pago le gustaría incluir en la aplicación?, transferencia y pago en efectivo
8. ¿Considera importante incluir promociones y ofertas especiales de sus productos en la aplicación?
9. ¿Le gustaría ofrecer opciones de entrega directa en el lugar de trabajo de los clientes?
10. ¿Considera importante contar con notificaciones del estado del pedido?

ANEXOS 4

Manual de Usuario

Manual de Usuario

Aplicación móvil Comedor UEB



Responsables:

Henry A. Ocaña
Lenon J. Ganchala

Versión:

1.0

Índice de Contenidos

Introducción14

Objetivo14

Alcance14

Restricciones14

GUIA PASO A PASO EN LA APLICACIÓN MOVIL15

1. ROL CLIENTE15

1.1. Acceder a la aplicación15

1.2. Registrar nuevo usuario16

1.3. Sección de productos16

1.4. Confirmar orden17

1.5. Pago20

2. Rol Restaurante22

2.1. Iniciar sesión22

2.2. Crear nueva categoría23

2.3. Crear nuevo producto24

2.4. Despachar Pedidos25

3. Rol Repartidor28

3.1. Iniciar sesión28

3.2. Entregar los pedidos28

Índice de Figuras

- Figura 1 Pantalla de Login15
- Figura 2 Pantalla de Registro16
- Figura 3 Pantalla de productos17
- Figura 4 Pasos para confirmar orden18
- Figura 5 Pasos para agregar la ubicación de entrega19
- Figura 6 Pasos para agregar el punto de referencia en el mapa de Google para crear mi dirección.20
- Figura 7 Pasos para realizar el pago21
- Figura 8 Pantalla de mis pedidos en el estado pago22
- Figura 9 Inicia sesión y pasa a la pantalla de los roles23
- Figura 10 Pasos para crear una nueva categoría24
- Figura 11 Pasos para crear un nuevo producto25
- Figura 12 Pasos para despachar una orden26
- Figura 13 Pasos para agregar repartidor y despachar la orden26
- Figura 14 Pasos para iniciar sesión como repartidor28
- Figura 15 Pasos para realizar la entrega de la orden29

Introducción

La aplicación móvil Comedor UEB de servicios de alimentación permite mejorar la gestión de pedidos y difusión de la información del menú, así como de sus promociones, esta aplicación fue diseñada para funcionar en la matriz de la Universidad Estatal de Bolívar buscando mejorar la experiencia de los clientes. Para conocer el funcionamiento de la aplicación el usuario puede guiarse en las instrucciones que se brinda en el manual, encontrara ilustraciones de los pasos a seguir para registrarse, iniciar sesión y realizar pedidos.

Objetivo

Proporcionar una navegabilidad guiada paso a paso sobre las diferentes secciones de la aplicación permitiendo aprovechar todas las funcionalidades que se ofrece.

Alcance

El presente documento proporciona una guía a todos los usuarios de la aplicación móvil Comedor UEB garantizando una apropiada usabilidad y funcionamiento, permitiendo una buena experiencia al momento de utilizarlo, aprovechando al máximo los servicios que ofrece.

Restricciones

Está diseñada con las siguientes características:

- Dispositivos móviles con sistema operativo Android
- Para el comedor de la matriz de la Universidad Estatal de Bolívar

GUIA PASO A PASO EN LA APLICACIÓN MÓVIL

1. ROL CLIENTE

1.1. Acceder a la aplicación

Para acceder a la aplicación debemos tener una cuenta registrada en la aplicación si es el caso podemos ingresar nuestros datos en los campos que nos indica como se muestra en la figura 1 y presionamos el botón de login.

Figura
Pantalla de Login

1



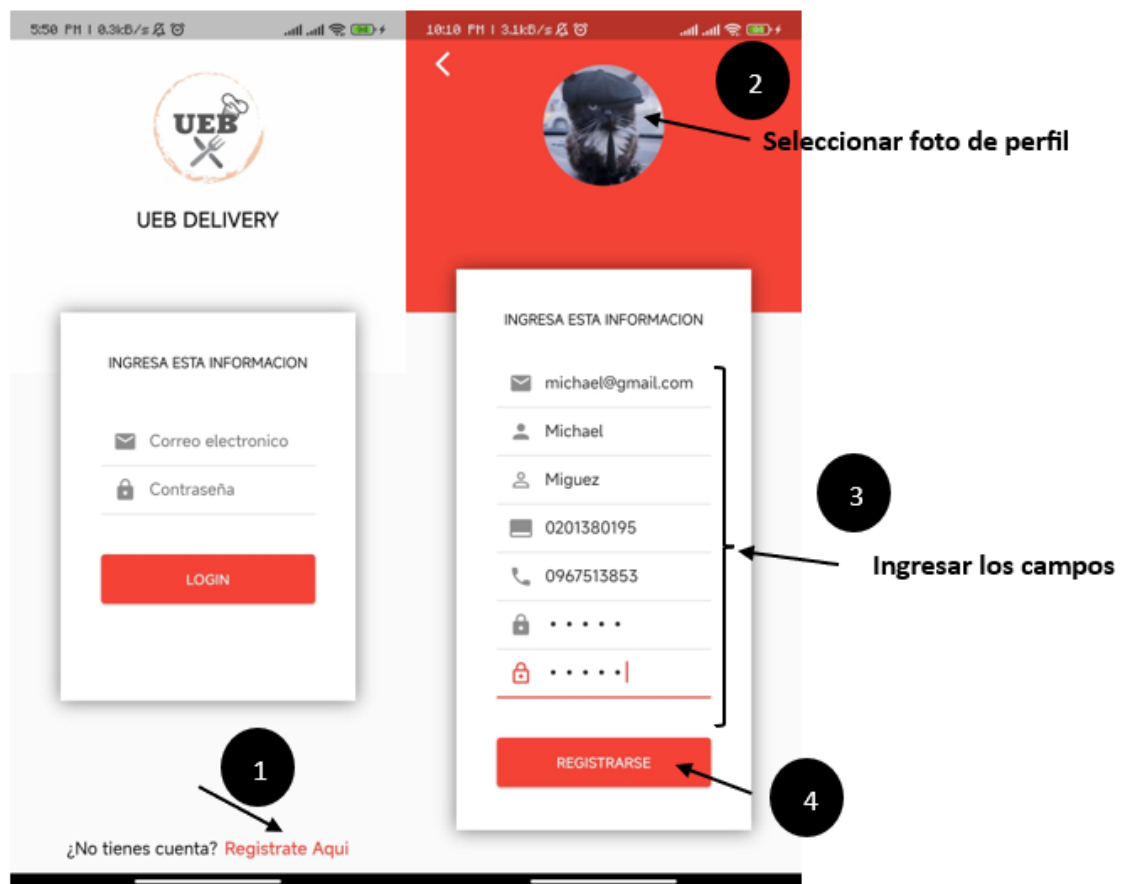
Ganchala L, Ocaña H. *Pantalla de login.*; 2023

1.2.Registrar nuevo usuario

Si no tienes una cuenta debes dirigirte a la parte inferior, ¿no tienes cuenta? Y seleccionar regístrate aquí como se muestra en la figura2 la cual nos lleva a la pantalla de registro en el que tenemos que seleccionar una imagen de perfil e ingresar los datos que nos pide.

Figura
Pantalla de Registro

2



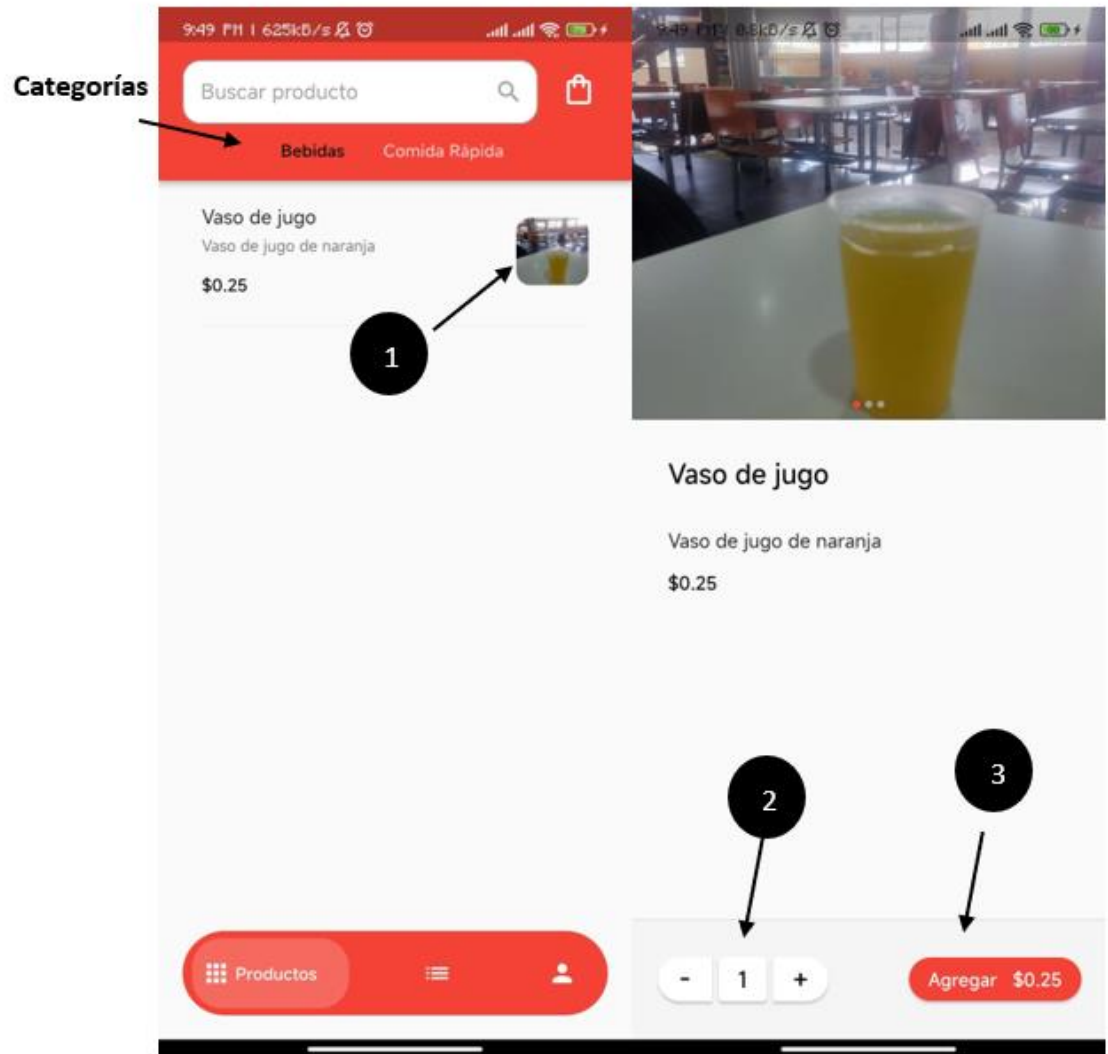
Ganchala L, Ocaña H. *Pantalla de registro.*; 2023.

1.3.Sección de productos

En la sección de productos se podrá visualizar los productos y las categorías, además cuenta con una barra de búsquedas para que encontrar un producto sea más eficiente. El cliente puede observar una pequeña imagen del producto con una breve descripción y el precio, el cual al presionar en el producto se desplaza una pantalla, en el que se puede apreciar mejor el producto, modificar la cantidad y agregar con se puede apreciar en la figura 3.

Figura
Pantalla de productos

3



Ganchala L, Ocaña H. *Pantalla de productos.*; 2023.

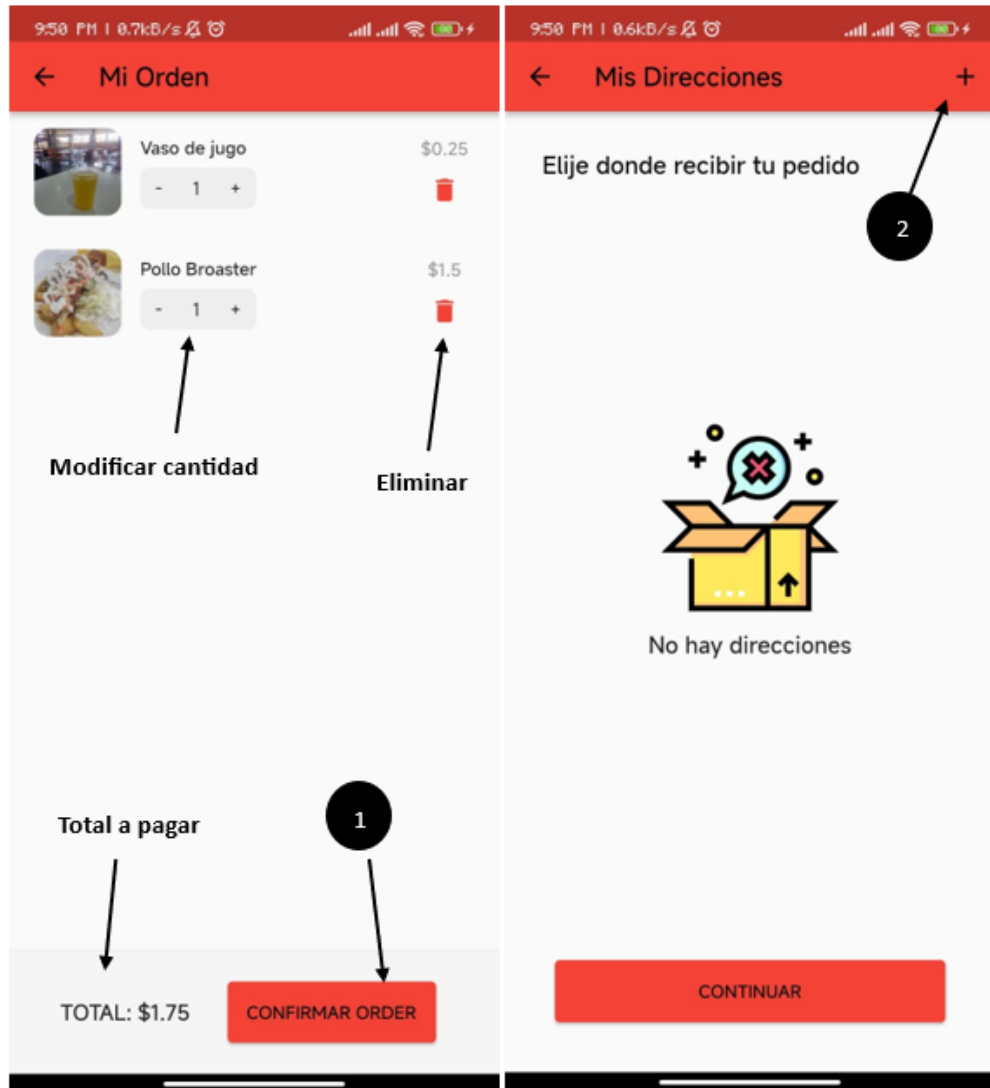
1.4. Confirmar orden

Una vez que presionamos el botón de agregar nos dirige a la pantalla de mi orden en el cual encontraremos los productos que hemos seleccionado, en el podemos modificar la cantidad, eliminar producto, también se muestra el total a pagar y el botón de confirmar, presionamos el botón el cual nos dirige a la pantalla de seleccionar ubicación, aquí agregaremos los datos de la dirección donde deseamos adquirir la orden, activamos el GPS para agregar la ubicación y presionamos el botón crear dirección, pasamos a la pantalla de mis direcciones

y seleccionamos la dirección que creamos y procedemos a presionar el botón continuar como se muestra en las siguientes figuras (4,5,6).

Figura
Pasos para confirmar orden

4



Ganchala L, Ocaña H. *Pantalla de confirmación de la orden.*; 2023

Figura
Pasos para agregar la ubicación de entrega

5

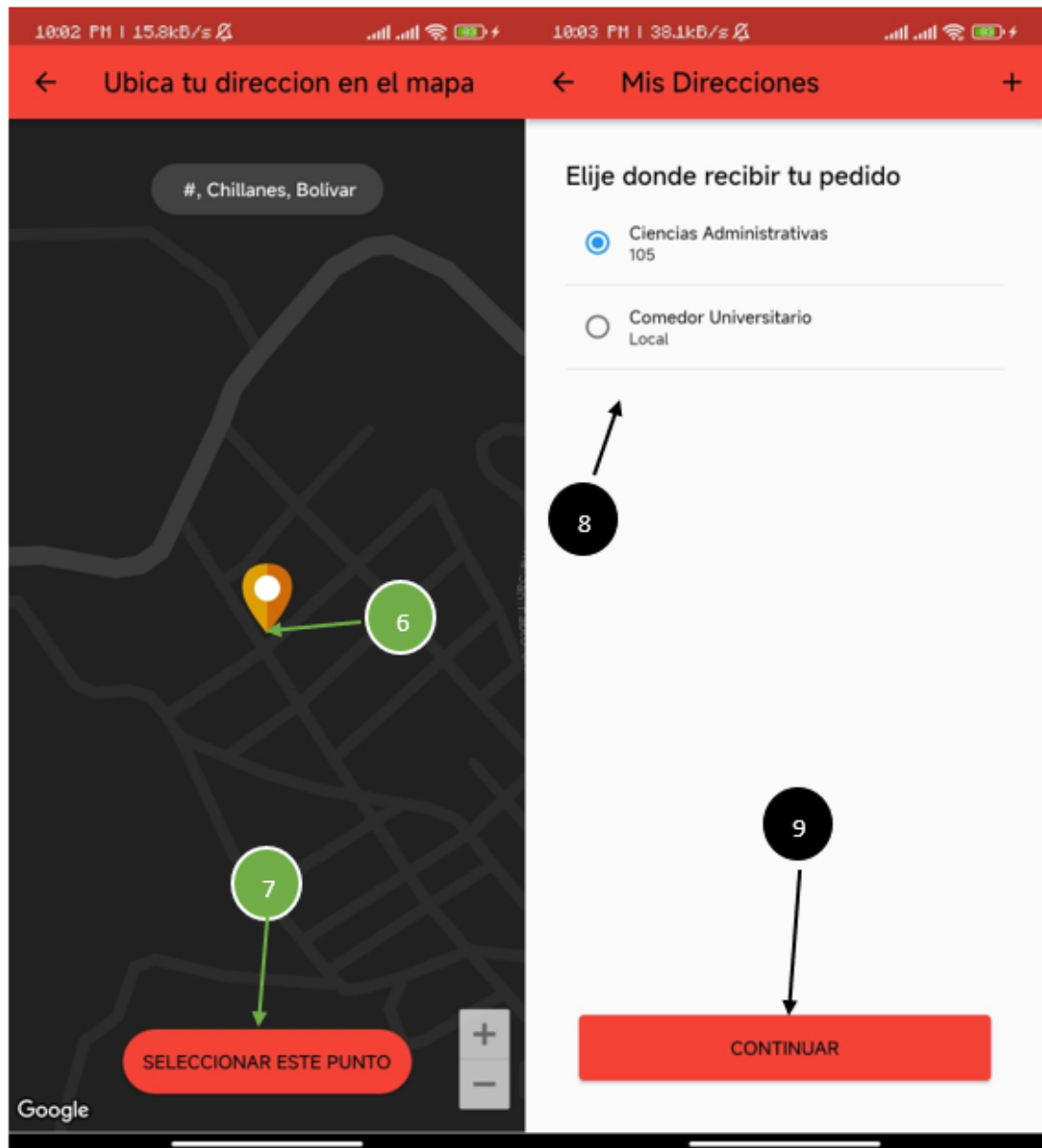


Ganchala L, Ocaña H. *Pantalla de direcciones.*; 2023.

Figura

6

Pasos para agregar el punto de referencia en el mapa de Google para crear mi dirección.



Ganchala L, Ocaña H. *Pantalla de registro de direcciones.*; 2023.

1.5.Pago

Luego de haber seleccionado el botón de continuar el cliente tiene que realizar el pago, entre ellos cuenta con transferencia y pago en efectivo. Al realizarlo por transferencia el usuario debe tomar captura del comprobante y subir la imagen, luego presiona el botón continuar y se muestra en pantalla los mensajes en la parte superior de proceso terminado y en la parte inferior la orden se ha

creado correctamente y se puede visualizar la orden en la sección de pedidos como se muestra en las figuras 7 y 8.

Figura
Pasos para realizar el pago

7



Ganchala L, Ocaña H. *Pantalla de pago.*; 2023

Figura
Pantalla de mis pedidos en el estado pago

8



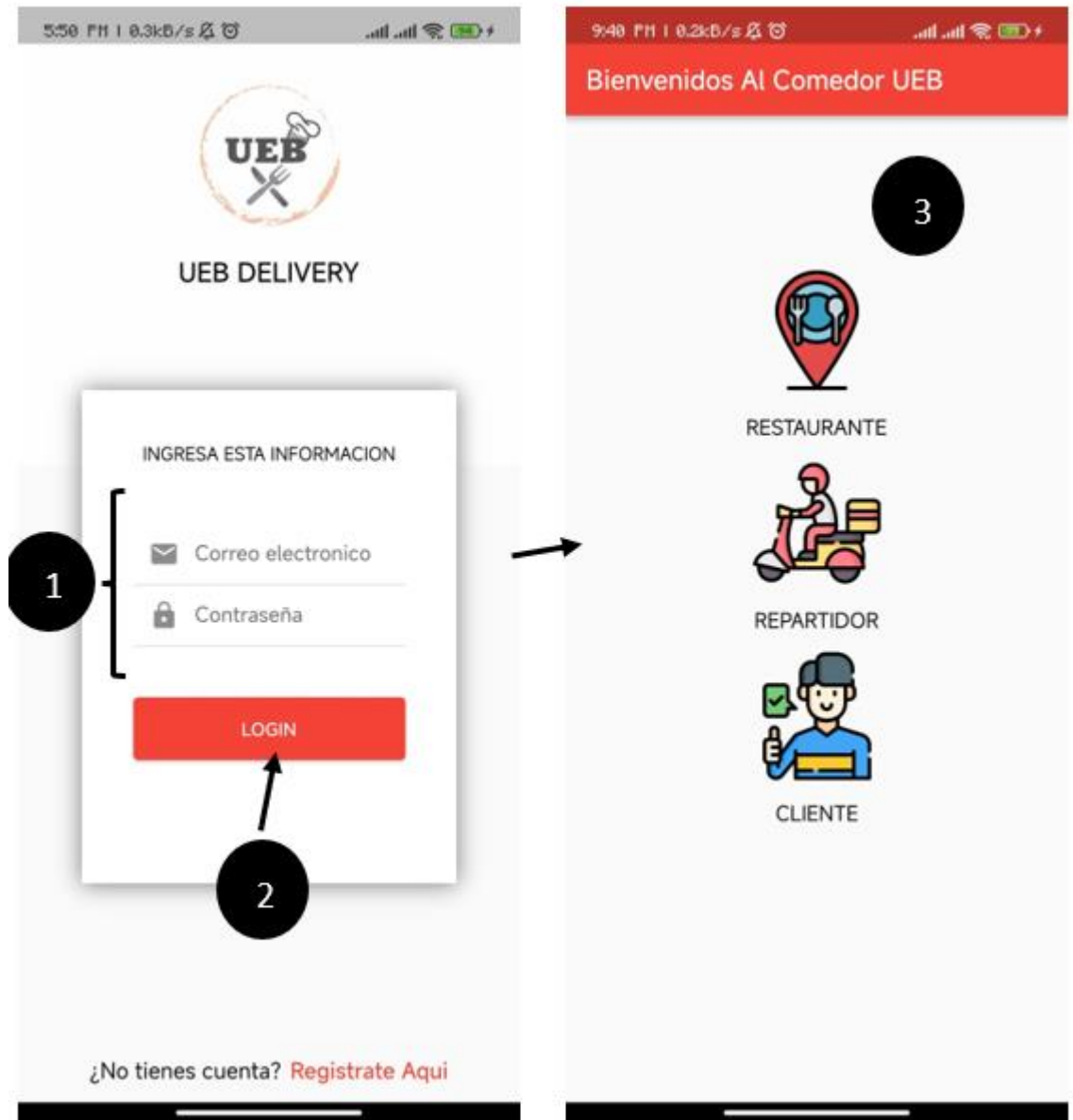
Ganchala L, Ocaña H. *Pantalla de pedidos.*; 2023.

2. Rol Restaurante

2.1. Iniciar sesión

El usuario debe ingresar las credenciales que le asignaron para poder ingresar a la aplicación y poder pasar a la pantalla de roles en el que podrá ingresar en cualquiera de los roles como se puede apreciar en la figura 9.

Figura
Inicia sesión y pasa a la pantalla de los roles

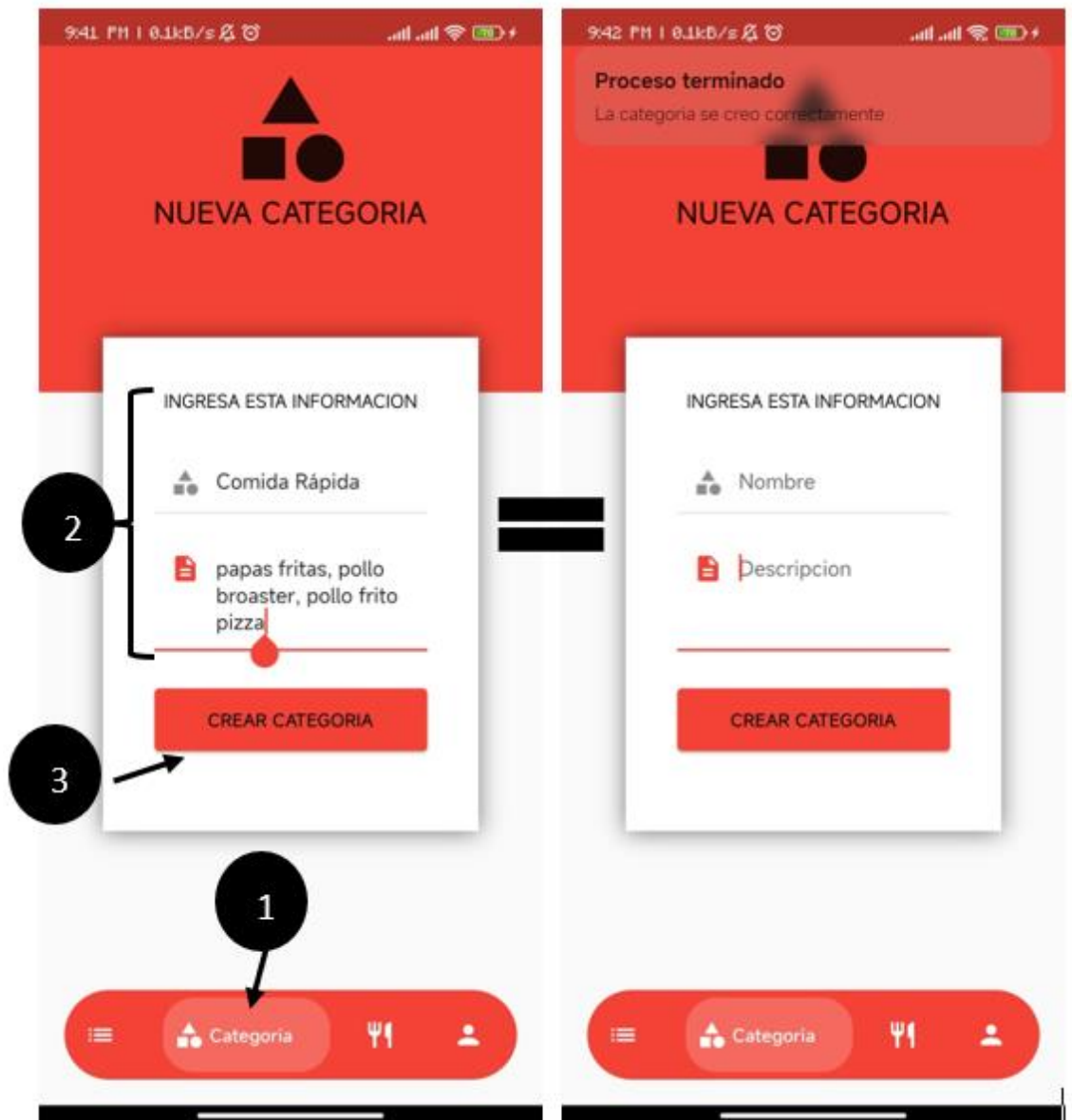


Ganchala L, Ocaña H. *Pantalla de roles.*; 2023.

2.2.Crear nueva categoría

Una vez ingrese a la categoría restaurante seleccione la sección categoría en el menú de la parte inferior, se desplegará una pantalla en la que el usuario puede ingresar los campos de nombre y descripción, luego debe presionar el botón crear categoría mostrando en pantalla proceso terminado y un mensaje que indica que la categoría se creó correctamente como se muestra en la figura

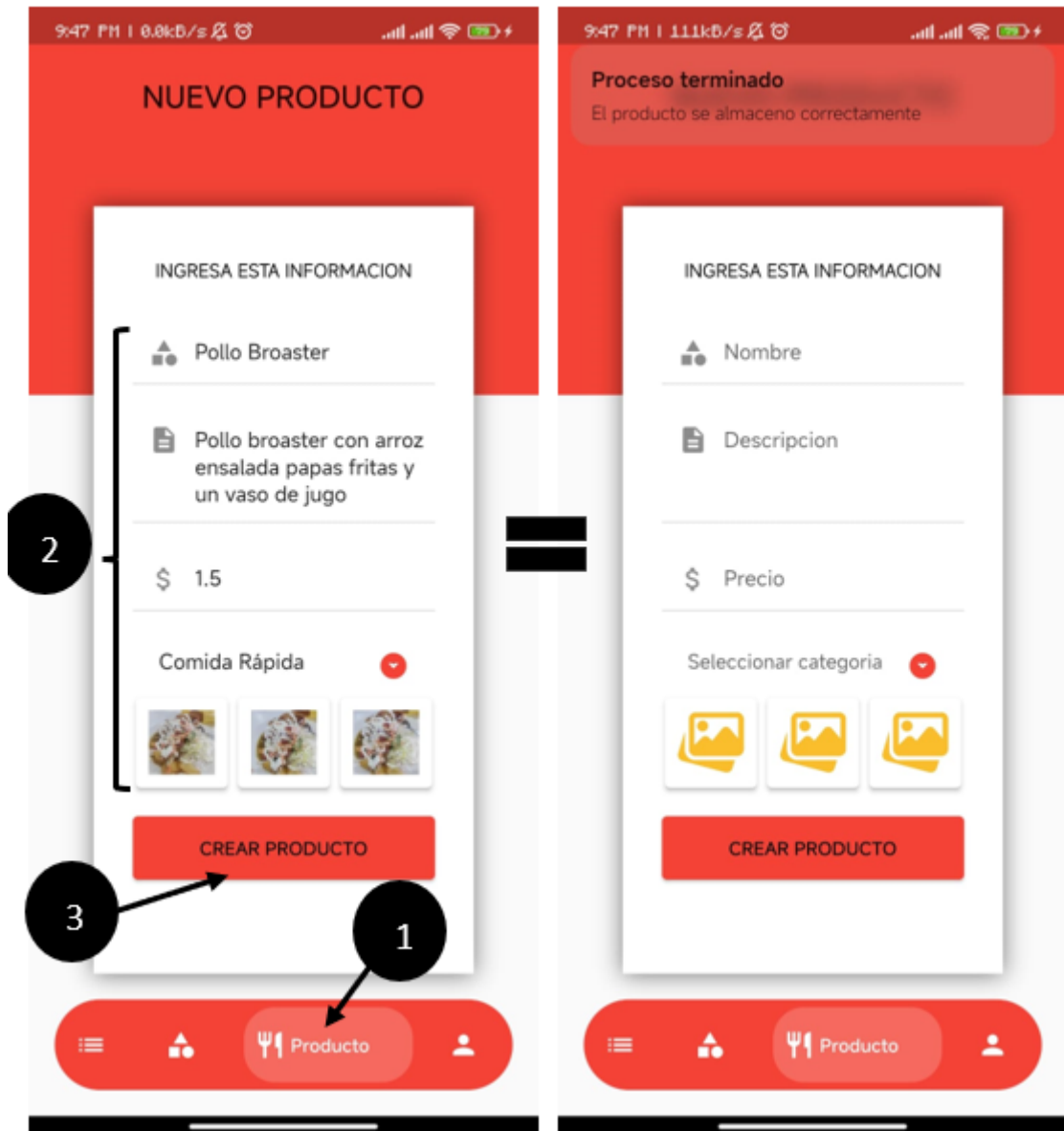
Figura
Pasos para crear una nueva categoría



Ganchala L, Ocaña H. *Pantalla de creación de categoría* .; 2023.

2.3.Crear nuevo producto

Para crear un nuevo producto debe ubicarse en la sección de producto y se despliega la pantalla para ingresar la información que tendrá el nuevo producto como nombre, descripción, precio, seleccionar la categoría a la que pertenece, seleccionar tres imágenes del producto, para luego presionar el botón de crear producto como se muestra en la figura 11.

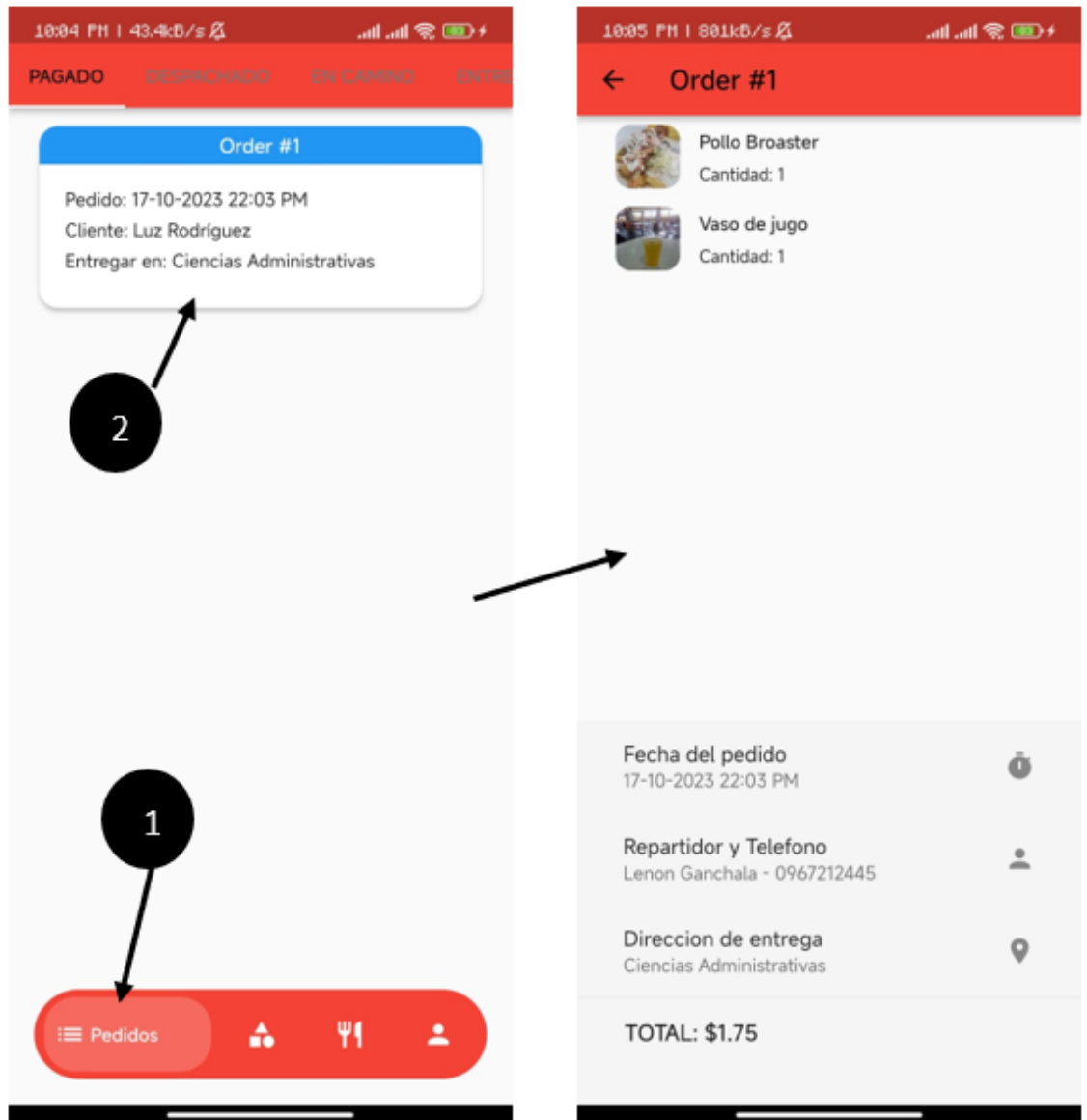


Ganchala L, Ocaña H. *Pantalla de creación de productos.*; 2023.

2.4.Despachar Pedidos

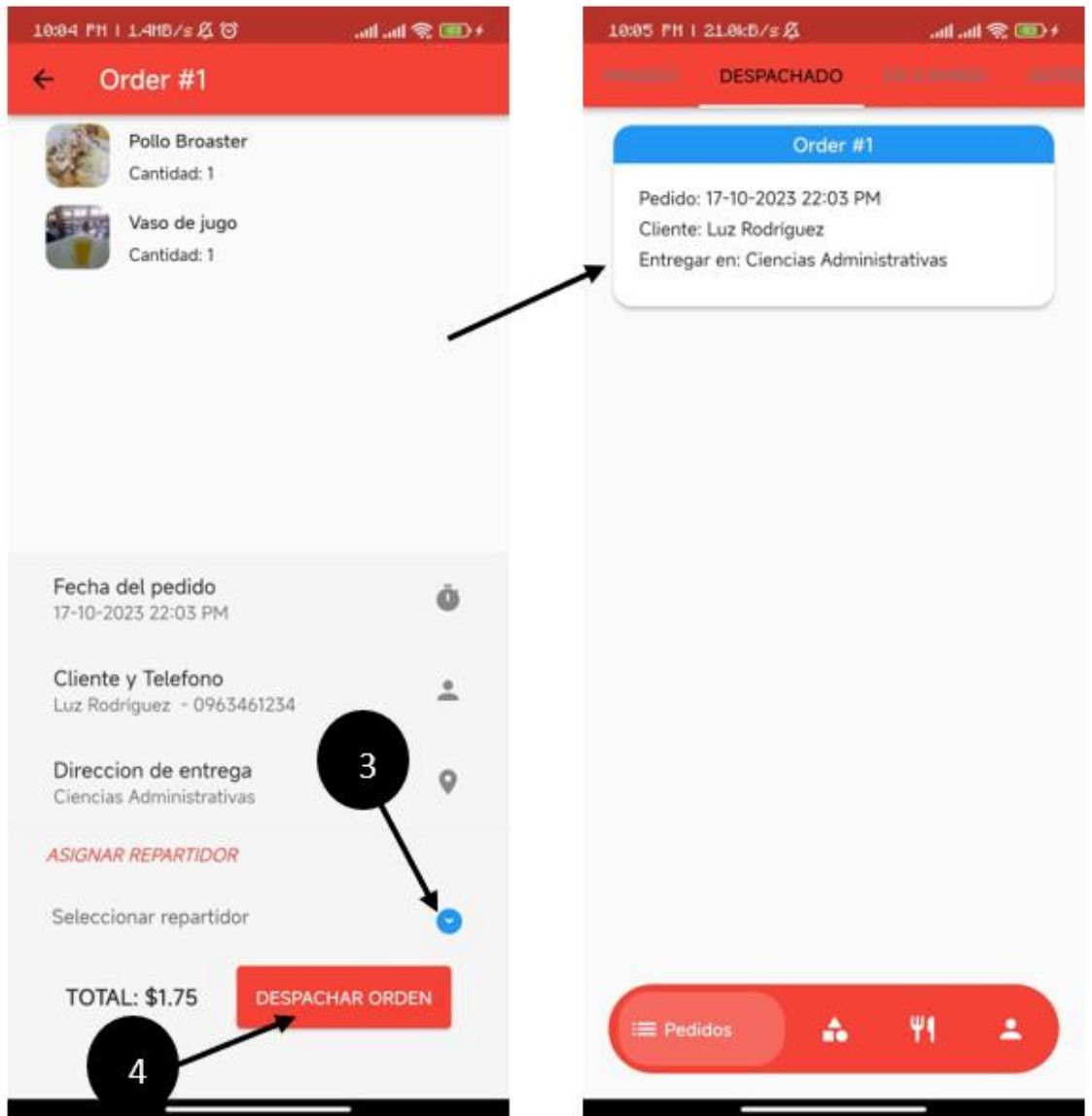
Para ingresar a los pedidos debe ingresar a la sección de pedidos en el menú de la parte inferior, para visualizar las órdenes en estado pagado para su preparación, debe presionar en la orden y se desplegara una pantalla en el que visualizara la información de los productos a preparar, después debe designar un repartidor y presionar el botón despachar orden, la cual pasa al estado despachado como se observa en la figura

Figura
Pasos para despachar una orden



Ganchala L, Ocaña H. *Pantalla de despacho de orden.*; 2023.

Figura 13
Pasos para agregar repartidor y despachar la orden



Ganchala L, Ocaña H. *Pantalla de despacho de orden por repartidor.*; 2023.

3. Rol Repartidor

3.1. Iniciar sesión

El usuario debe ingresar la cuenta asignada al rol repartidor y la contraseña en los campos y después presionar el botón login para ingresar a la aplicación como se muestra en la figura 14

Figura 14
Pasos para iniciar sesión como repartidor



Ganchala L, Ocaña H. *Pantalla de login.*; 2023.

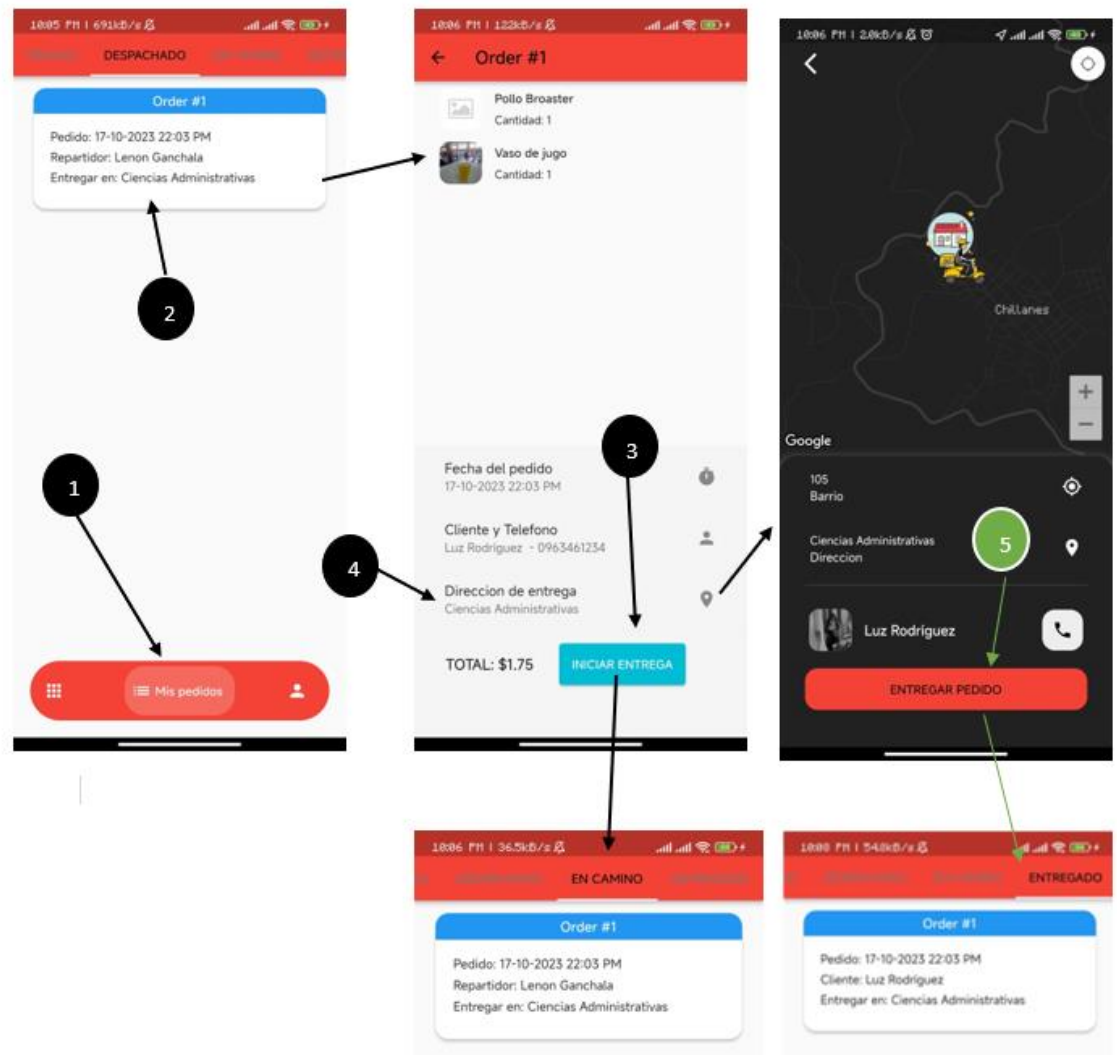
3.2. Entregar los pedidos

El usuario debe ingresar a la sección de pedidos que se encuentra en el menú de la parte inferior en el cual se le despliega la pantalla de pedidos, después debe seleccionar la orden la cual desplegara la información del pedido y presionar el botón de iniciar entrega, pasando la orden al estado encamino. Para guiarse al lugar de destino debe presionar dirección de entrega para desplegar el mapa de Google, en él se aprecia un icono de repartidor y la ruta al destino, en el que se presentara un icono de casa cuando esté cerca del

punto a entregar, luego puede presionar el botón de entregar pedido para pasar la orden al estado de entregado. Además, el usuario tiene la funcionalidad de ponerse en contacto con el cliente de la orden, el cual se encuentra antes del botón entregar pedido. Como se muestra en las figuras

Figura Pasos para realizar la entrega de la orden

15



Ganchala L, Ocaña H. *Pantalla de entrega.*; 2023.

ANEXOS 5

Manual Técnico

Manual Técnico



Aplicación móvil del comedor universitario

Lenon G. Joel & Henry O. Alexander
Código: U-01
Versión: 1.0
Fecha: 18-10-2023
Nivel Confidencialidad: Publico

Índice

Introducción	3
Instalación y configuración del sistema	5
Controllers.....	5
FrontEnd Estructura en Android Estudio	8
Clase Main	9
El pubspec.yaml	11

Introducción

Este documento proporciona una descripción general de las tecnologías y pre-requisitos necesarios para desarrollar la aplicación móvil del comedor universitario de la Universidad Estatal de Bolívar. Estos requisitos son esenciales para garantizar el correcto funcionamiento de la aplicación y permitir futuras actualizaciones y mantenimiento.

Pre-requisitos de instalación de la base de datos, del framework y sus dependencias

Node.js versión v18.18.1 o superior

Flutter 3.13.6 • channel stable o superior

Api Google fire base o superior

mysql-8.1.0 o superior

Herramientas empleadas en el desarrollo

Node.js: es un entorno diseñado para aplicaciones que pueden gestionar eficientemente múltiples conexiones simultáneas sin enfrentar bloqueos de procesos, gracias a su arquitectura basada en eventos.

Flutter: es un conjunto de herramientas desarrollado por Google para la creación de aplicaciones móviles en diferentes plataformas, lo que incluye sistemas Android y iOS de Apple. Flutter sobresale como uno de los proyectos de desarrollo de aplicaciones móviles con un crecimiento destacado. No solo se limita a dispositivos móviles, ya que, desde la última versión estable, también se ha expandido su alcance para incluir aplicaciones web y aplicaciones de escritorio, compatibles con sistemas Windows y Mac. Cabe mencionar que estas funcionalidades en las plataformas de escritorio se encuentran en una fase experimental (beta).

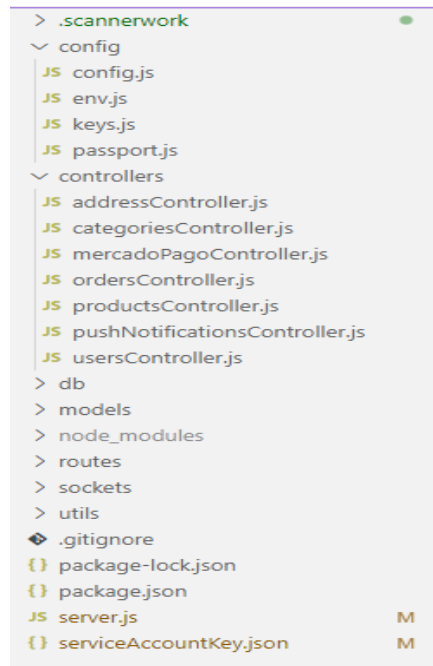
Firebase: es la plataforma de Google que hace que la creación de aplicaciones web y móviles de alta calidad sea eficiente y rápida.

Git: es un proyecto de código abierto maduro y mantenido activamente desarrollado. Una sorprendente cantidad de proyectos de software dependen de Git para el control de versiones, incluidos proyectos comerciales y de código abierto.

Android Studio: es el entorno de desarrollo integrado (IDE) oficial que se utiliza para desarrollar aplicaciones de Android. Basado en el potente editor de código y las herramientas para desarrolladores de IntelliJ IDEA, Android Studio ofrece aún más funciones para crear aplicaciones.

Visual Studio Code (también conocido como VS Code) es un editor de código fuente desarrollado por Microsoft para Windows, Linux, macOS y la web. Incluye soporte de depuración, administración de Git incorporada, resaltado de sintaxis, finalización inteligente de código, fragmentos y re factorización de código.

Instalación y configuración del sistema



- Ingresar a la ruta /backendlivery/ para acceder al rar código fuente
- tenemos la siguiente estructura dentro del rar

```
config > JS config.js > ...
1  const mysql = require('mysql');
2
3  const db = mysql.createConnection({
4    host: 'localhost',
5    user: 'root',
6    password: 'password',
7    database: 'UebDelivery'
8  });
9
10 db.connect(function(err) {
11   if (err) throw err;
12   console.log('DATABASE CONNECTED!');
13 });
14
15 module.exports = db;
```

en el archivo config.js estará la conexión a la base de datos del Backend

Ganchala L, Ocaña H. *Estructura del backend.*; 2023.

Controllers

keys.js

Se encontrará la clave para cifrar las contraseñas de los usuarios

passport

Importa las dependencias necesarias, incluyendo JwtStrategy y ExtractJwt del paquete passport-jwt, así como otros módulos como Keys (que probablemente contiene secretos o claves) y User (que parece ser un modelo de usuario).

Exporta una función que toma un objeto passport como argumento. Esto sugiere que se trata de un módulo de configuración de autenticación que utiliza Passport.js.

Dentro de la función exportada, se configura una estrategia de autenticación JWT. Se crea un objeto opts que contiene la configuración para la estrategia JWT.

Se especifica la fuente de los tokens JWT en opts.jwtFromRequest utilizando ExtractJwt.fromAuthHeaderWithScheme('jwt'), lo que significa que los tokens se esperan en el encabezado de autorización con el esquema "jwt".

La clave secreta que se utilizará para verificar la firma de los tokens JWT se almacena en opts.secretOrKey, que se obtiene desde el módulo Keys.

Luego, se crea y configura la estrategia de autenticación JWT con passport.use(). Esta estrategia verifica y decodifica los tokens JWT en las solicitudes de autenticación.

Dentro de la estrategia, se llama a User.findById() para buscar un usuario en la base de datos basado en el jwt_payload.id. Si se encuentra un usuario, se llama a done (null, user) para indicar que la autenticación es exitosa. Si hay un error, se llama a done (err, false) y si no se encuentra ningún usuario, se llama a done (null, false) para indicar que la autenticación falló.

addressController

Se encarga de controlar la dirección de los clientes que se almacenen en la base de datos

categoriesController

Se encarga de controlar las categorías de la aplicación que se almacenen en la base de datos

ordersController

Se encarga de controlar las órdenes del cliente que se almacenen en la base de datos

productsController

Se encarga de controlar los productos que se almacenen en la base de datos

pushNotificationsController

Se encarga de controlar las notificaciones y de que se distribuya de acuerdo a su id

usersController

Controlamos a los usuarios que estén almacenados en la

db.sql

```
USE UebDelivery;

CREATE TABLE users(
  id BIGINT PRIMARY KEY AUTO_INCREMENT,
  email VARCHAR(180) NOT NULL UNIQUE,
  name VARCHAR(90) NOT NULL,
  lastname VARCHAR(90) NOT NULL,
  phone VARCHAR(90) NOT NULL UNIQUE,
  image VARCHAR(255) NULL,
  password VARCHAR(90) NOT NULL,
  created_at TIMESTAMP(0) NOT NULL,
  updated_at TIMESTAMP(0) NOT NULL
);

CREATE TABLE roles(
  id BIGINT PRIMARY KEY AUTO_INCREMENT,
  name VARCHAR(90) NOT NULL UNIQUE,
  image VARCHAR(255) NULL,
  route VARCHAR(180) NOT NULL,
  created_at TIMESTAMP(0) NOT NULL,
  updated_at TIMESTAMP(0) NOT NULL
);

INSERT INTO roles(
  name,
  route,
  created_at,
  updated_at
)
VALUES(
  'RESTAURANTE',
  '/restaurant/orders/list',
  '2022-03-27',
  '2022-03-27'
```

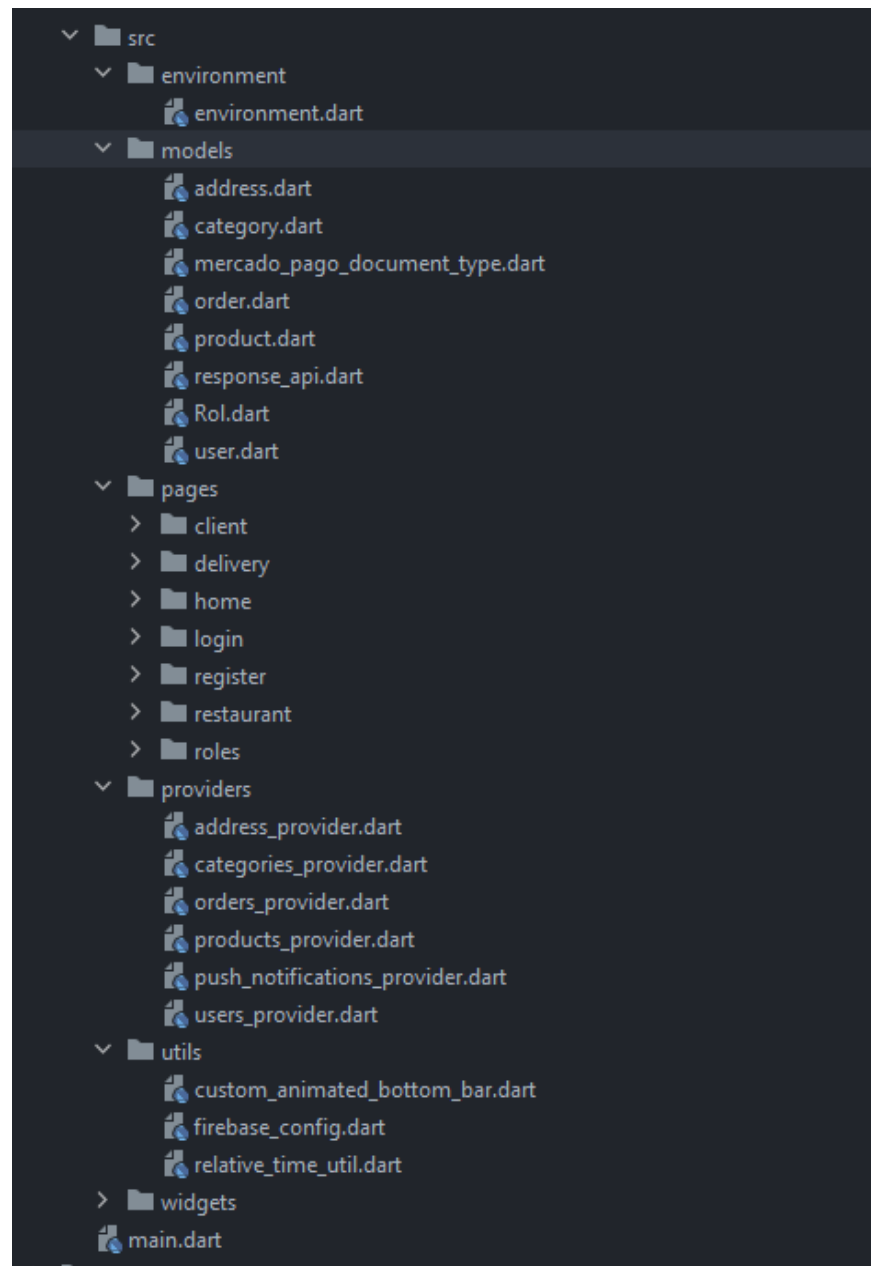
Se encontrará todas las entradas para restablecer la base de datos para

```
.\BackendDeliveryMySQL> node server.js
```

levantar el sistema se necesita ejecutar el siguiente código:

Ganchala L, Ocaña H. *Estructura de la base de datos.*;2023.

FrontEnd Estructura en Android Estudio



Ganchala L, Ocaña H. *Estructura del front end.*; 2023.

Environment

Aquí están todas las conexiones para la poderse comunicar con el back end

Models

En esta carpeta tenemos los objetos que se utilizan en todas las clases para utilizar los métodos de nuestras páginas. Se enlaza especialmente con las páginas y los controles que tenemos

Pages

Están todas las páginas que se muestran en la aplicación, así como sus respectivos controles, que dan la visualización a la cada parte de la aplicación

Providers

Se encargan de ubicar las clases para poder formar el control respectivo con los controladores de las páginas para poder enviar datos al BackEnd

Utils

La clase de configuraciones y entornos de animación y configuración del Fire Base de Google

Widgets

Encargados del tiempo de ejecución de la aplicación

Clase Main

```
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.core.view.WindowCompat;
import androidx.fragment.app.Fragment;
import androidx.lifecycle.ViewModelProvider;
import androidx.navigation.Navigation;
import androidx.navigation.ui.AppBarConfiguration;
import androidx.navigation.ui.NavigationUI;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import androidx.appcompat.widget.Toolbar;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.core.view.WindowCompat;
import androidx.fragment.app.Fragment;
import androidx.lifecycle.ViewModelProvider;
import androidx.navigation.Navigation;
import androidx.navigation.ui.AppBarConfiguration;
import androidx.navigation.ui.NavigationUI;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import androidx.appcompat.widget.Toolbar;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.core.view.WindowCompat;
import androidx.fragment.app.Fragment;
import androidx.lifecycle.ViewModelProvider;
import androidx.navigation.Navigation;
import androidx.navigation.ui.AppBarConfiguration;
import androidx.navigation.ui.NavigationUI;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import androidx.appcompat.widget.Toolbar;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.core.view.WindowCompat;
import androidx.fragment.app.Fragment;
import androidx.lifecycle.ViewModelProvider;
import androidx.navigation.Navigation;
import androidx.navigation.ui.AppBarConfiguration;
import androidx.navigation.ui.NavigationUI;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

public class MainActivity extends AppCompatActivity implements AppBarConfiguration.OnAppBarStateChangedListener {
    private AppBarConfiguration appBarConfiguration;
    private NavController navController;
    private Toolbar toolbar;
    private RecyclerView recyclerView;
    private LinearLayoutManager layoutManager;
    private RecyclerView.Adapter adapter;
    private FirebaseFirestore database;
    private FirebaseAuth firebaseAuth;
    private FirebaseAuth.AuthResultListener mAuthListener;

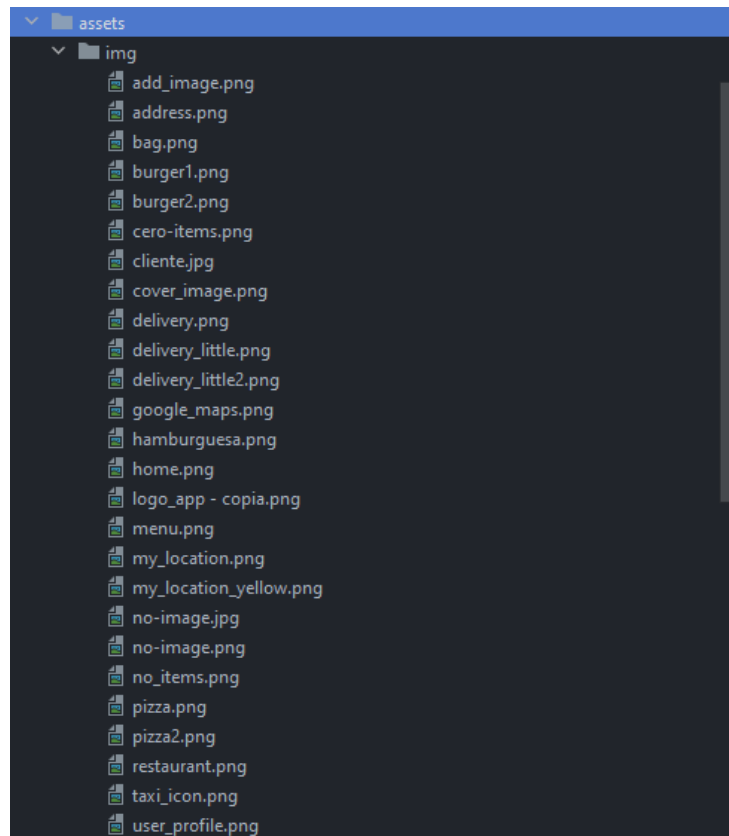
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        setSupportActionBar(toolbar);
        recyclerView.setLayoutManager(layoutManager);
        recyclerView.setAdapter(adapter);
        database = FirebaseFirestore.getInstance();
        firebaseAuth = FirebaseAuth.getInstance();
        mAuthListener = new FirebaseAuth.AuthResultListener() {
            @Override
            public void onAuthStateChanged(@NonNull FirebaseAuth firebaseAuth) {
                if (firebaseAuth.getCurrentUser() != null) {
                    // User is logged in
                } else {
                    // User is not logged in
                }
            }
        };
        firebaseAuth.addAuthStateListener(mAuthListener);
    }

    @Override
    public void onAppBarStateChanged(AppBarConfiguration appBarConfiguration) {
        // AppBar configuration change handler
    }
}
```

Ganchala L, Ocaña H. Estructura de la clase main.; 2023.

La clase en la cual se complementan todas las clases la rama principal de la aplicación en ella se listan nuevas paginas al igual que el tema principal en cuestión de color y colores para la aplicación

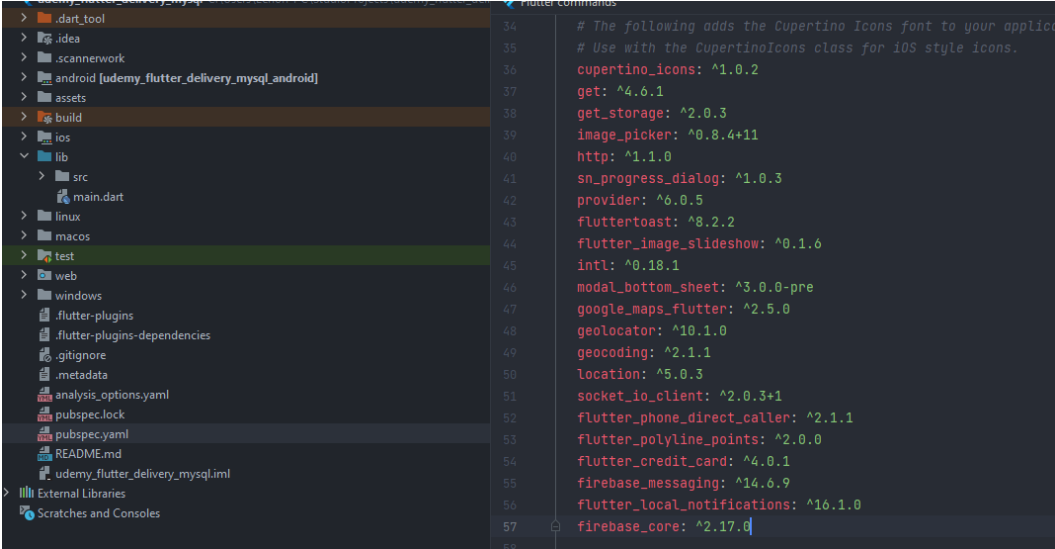
La carpeta Assets



Ganchala L, Ocaña H. *Directorio de las imágenes.*; 2023.

Tiene todas las imágenes que son utilizadas dentro del proyecto de la aplicación

El pubspec.yaml



```
34 # The following adds the Cupertino Icons font to your applic
35 # Use with the CupertinoIcons class for iOS style icons.
36/cupertino_icons: ^1.0.2
37/get: ^4.6.1
38/get_storage: ^2.0.3
39/image_picker: ^0.8.4+11
40/http: ^1.1.0
41/sn_progress_dialog: ^1.0.3
42/provider: ^6.0.5
43/fluttertoast: ^8.2.2
44/flutter_image_slideshow: ^0.1.6
45/intl: ^0.18.1
46/modal_bottom_sheet: ^3.0.0-pre
47/google_maps_flutter: ^2.5.0
48/geolocator: ^10.1.0
49/geocoding: ^2.1.1
50/location: ^5.0.3
51/socket_io_client: ^2.0.3+1
52/flutter_phone_direct_caller: ^2.1.1
53/flutter_polyline_points: ^2.0.0
54/flutter_credit_card: ^4.0.1
55/firebase_messaging: ^14.6.9
56/flutter_local_notifications: ^16.1.0
57/firebase_core: ^2.17.0
```

Ganchala L, Ocaña H. *Estructura del plugins de flutter.*; 2023.

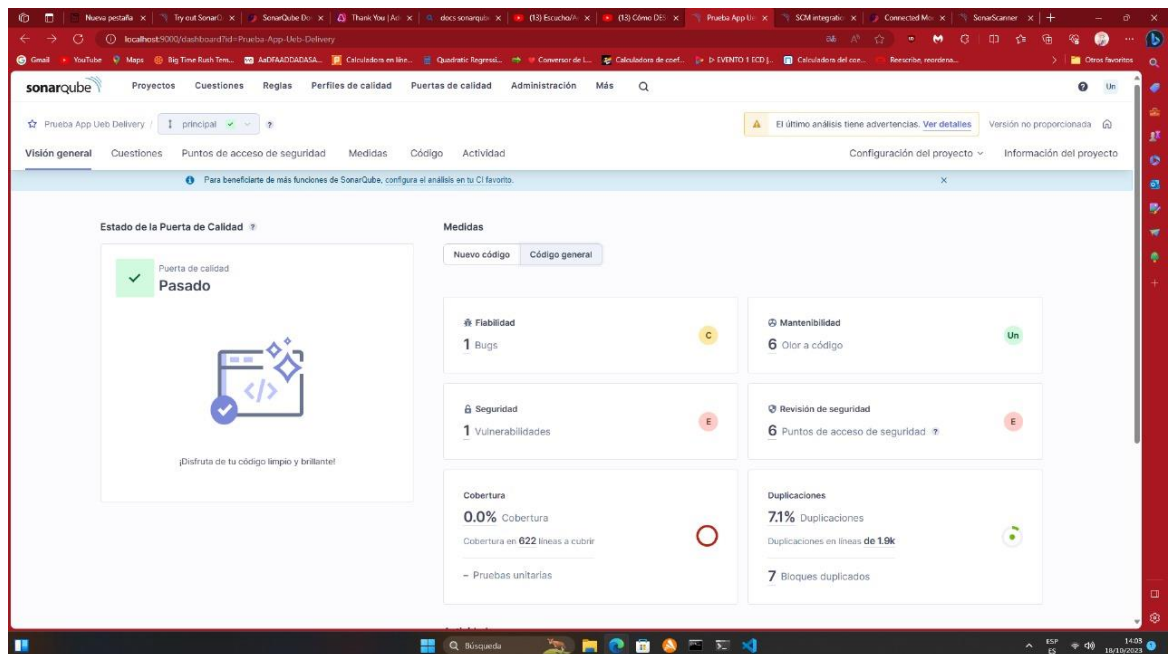
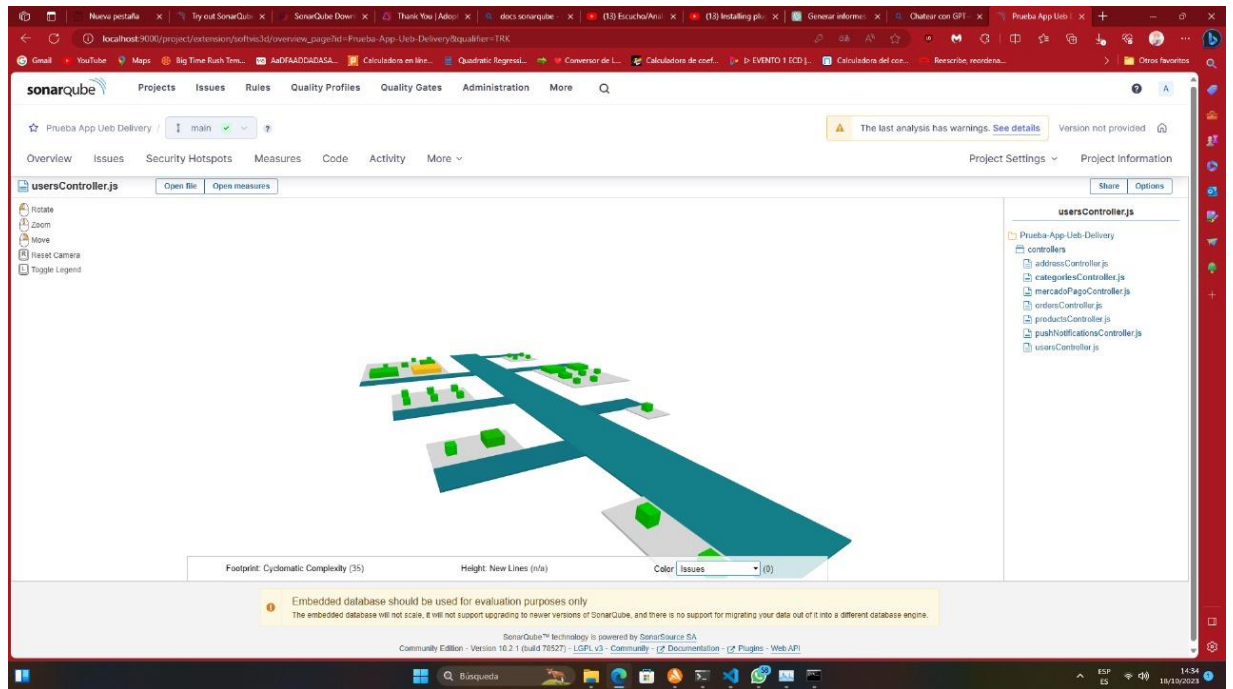
Es el encargado de incluir librerías si estas que se aprecia en la imagen no están el proyecto no podría ejecutarse es muy importante que se pueda ver la compatibilidad caso contrario se debería de poner las mismas librerías de los distintos paquetes

ANEXOS 6

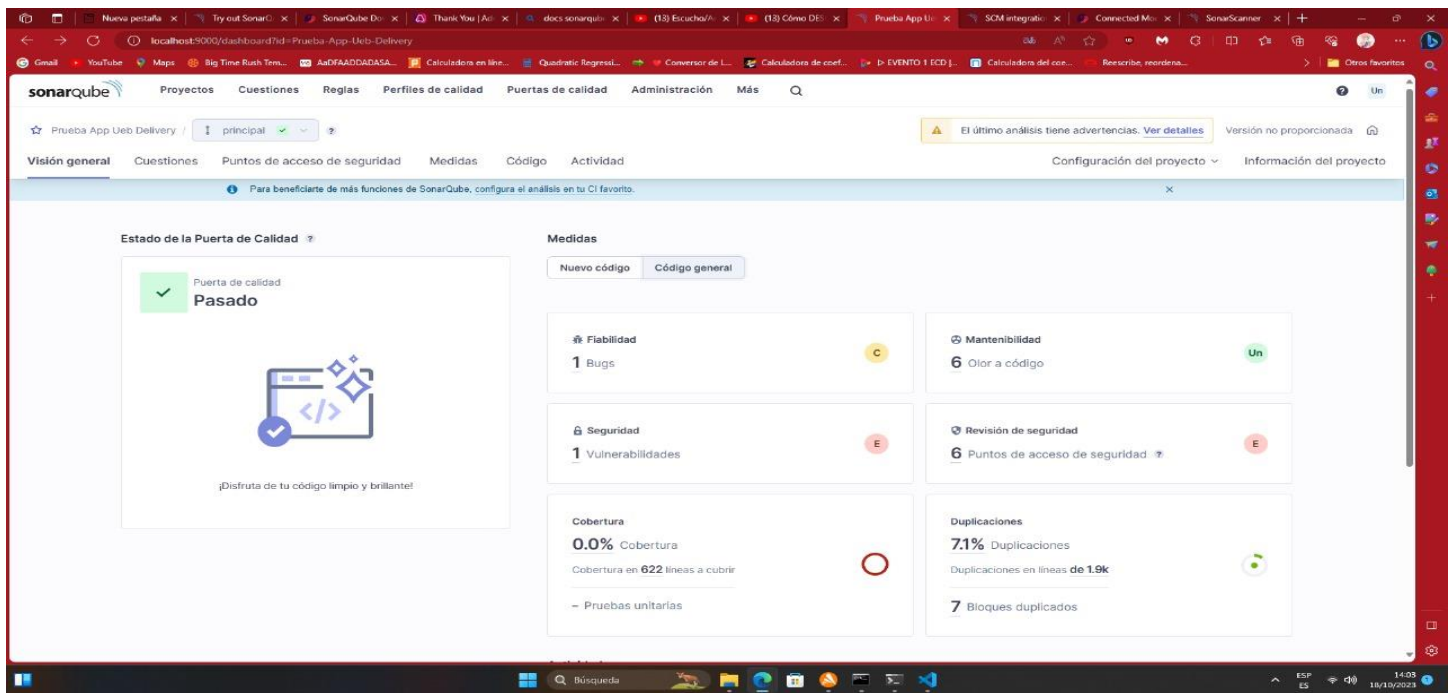
Ejecución de las Pruebas

Pruebas realizadas con la herramienta sonarQube

Esta herramienta realiza un análisis estático del código la cual nos permitió realizar un análisis del código, pruebas de seguridad, detección de código duplicado, entre otras.

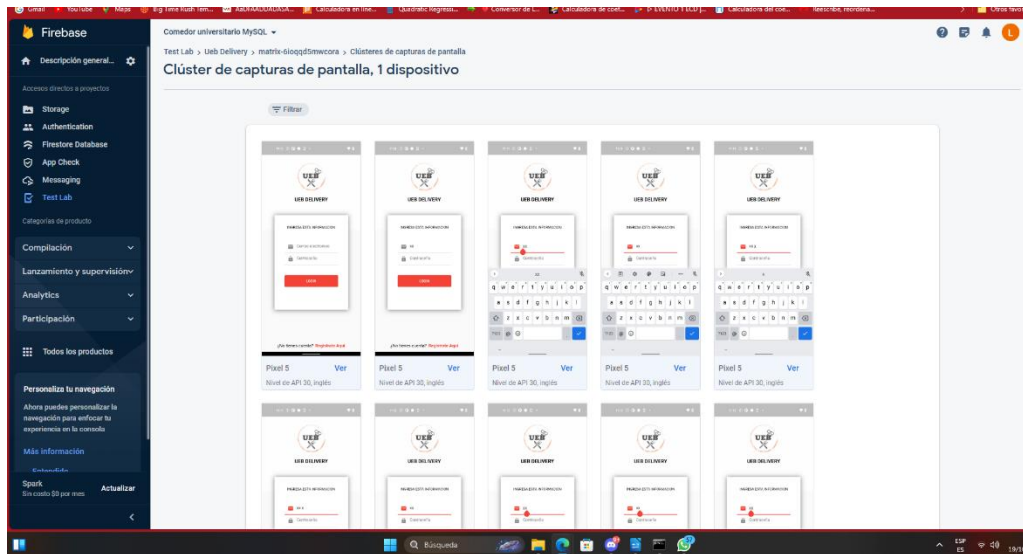


Ganchala L, Ocaña H. *Pruebas SonarQube*; 2023.



Ganchala L, Ocaña H. *Pruebas SonarQube*; 2023

Prueba de robo



Ganchala L, Ocaña H. *Pruebas RoboTest*; 2023

Robotest es una herramienta de prueba integrada en Firebase Test Lab¹. Esta herramienta analiza la estructura de la interfaz de usuario de su aplicación, la examina sistemáticamente y simula automáticamente el comportamiento del usuario. Robotest es una prueba automatizada que analiza la interfaz de usuario de su aplicación y luego la explora metódicamente simulando las acciones del usuario sin escribir ningún código.

Robotest captura prototipos, guarda una serie de capturas de pantalla comentadas y crea un vídeo a partir de las capturas de pantalla que muestra las acciones simuladas del usuario. Estos registros, capturas de pantalla y videos lo ayudarán a determinar la causa del bloqueo de sus aplicaciones. Estas funciones de prueba ocultas también lo ayudan a encontrar problemas en la interfaz de usuario de su aplicación¹.

Además de ejecutar pruebas Robo estándar, puede personalizar sus pruebas con scripts Robo, una característica de las pruebas Robo. Consulte Ejecutar un Robo Script para obtener más información. Si desea probar la versión beta de Robo iOS, consulte Ejecutar la prueba de Robo.

Prueba de encriptado:

Donde podemos observar la encriptación de la clave y de su asignación de **password** con su validación en firebase

```
Usuario obtenido: RowDataPacket {
  id: 1,
  email: 'lenonjoel123@gmail.com',
  name: 'lenon',
  lastname: 'ganchala',
  cedula: '02000608913',
  image: 'https://firebasestorage.googleapis.com/v0/b/comedor-universitario-mysql.appspot.com/o/image_1609895564054?alt=media&token=c53e4f86-a05f-435a-bfe7-ca0401319218',
  phone: '0967212445',
  password: '$2a10$FyPpaqjyVomkSt3AfG5eueLvtlyAV6R9ebCD65GosxzdQNTiYg',
  roles: [{"id": "1", "name": "RESTAURANTE", "image": "https://firebasestorage.googleapis.com/v0/b/comedor-universitario-mysql.appspot.com/o/restaurante.png?alt=media&token=c855026f-c423-4261-8cc1-c7c717c85d67", "route": "/restaurant/home"}, {"id": "2", "name": "REPARTIDOR", "image": "https://firebasestorage.googleapis.com/v0/b/comedor-universitario-mysql.appspot.com/o/repartidor.png?alt=media&token=c7d4ae8-f401-418c-949a-d587e24c4fd1", "route": "/delivery/home"}, {"id": "3", "name": "CLIENTE", "image": "https://firebasestorage.googleapis.com/v0/b/comedor-universitario-mysql.appspot.com/o/servicio-al-cliente.png?alt=media&token=e78fba38-6fce-4e9f-b949-662612ec66e7", "route": "/client/home"}]
}
```

Ganchala L, Ocaña H. *Pruebas de encriptado*; 2023

ANEXOS 7

Evidencias asistidas por la directora y los pares.













ANEXOS 8

Certificado Antiplagio

**ING. MÓNICA ELIZABETH BONILLA MANOBANDA EN CALIDAD DE
DIRECTORA DEL TRABAJO DE INTEGRACIÓN CURRICULAR,**

CERTIFICA

Que el trabajo de integración curricular denominado " APLICACIÓN MÓVIL PARA LA GESTIÓN EFICIENTE DE PEDIDOS EN LOS LOCALES DEL COMEDOR UNIVERSITARIO DE LA UNIVERSIDAD ESTATAL DE BOLÍVAR, AÑO 2023.", presentado por LENON JOEL GANCHALA CRUZ Y HENRY ALEXANDER OCAÑA PEÑA estudiantes de la Carrera de Software pasó el análisis de coincidencia no accidental en la herramienta Turnitin, reflejando un porcentaje de similitud del 8%, como se puede evidenciar en el documento adjunto.

Guaranda, 19 de octubre del 2023

Atentamente,



Ing. Mónica Bonilla M.

Directora

NOMBRE DEL TRABAJO

Trabajo de titulación definitivo.docx

AUTOR

LENON JOEL GANCHALA CRUZ

RECuento DE PALABRAS

15066 Words

RECuento DE CARACTERES

91113 Characters

RECuento DE PAGINAS

158 Pages

TAMANO DEL ARCHIVO

4.8MB

FECHA DE ENTREGA

Oct 19, 2023 7:24 PM GMT-5

FECHA DEL INFORME

Oct 19, 2023 7:26 PM GMT-5

● **8% de similitud general**

El total combinado de todas las coincidencias, incluidas las fuentes superpuestas, para cada base

- 7% Base de datos de Internet
- Base de datos de Crossref
- 6% Base de datos de trabajos entregados
- 2% Base de datos de publicaciones
- Base de datos de contenido publicado de Crossref

● **Excluir del Reporte de Similitud**

- Material bibliografico
- Material citado
- Fuentes excluidas manualmente
- Material citado
- Coincidencia baja (menos de 13 palabras)

ANEXOS 9

Link del repositorio digital de biblioteca donde fue subido el proyecto