Proyecto Final

Presentado por Grupo 6:

Ana María González Hernández - *anagonzalezhe@unal.edu.co*
Daniel Felipe Soracipa - *dsoracipa@unal.edu.co*
Juan José Medina Guerrero - *jmedinagu@unal.edu.co*
Samuel Josué Vargas Castro - *samvargasca@unal.edu.co*

Profesor:
Oscar Eduardo Alvarez Rodriguez
oalvarezr@unal.edu.co

Enero 31 de 2025

Universidad Nacional de Colombia
Facultad de Ingeniería
Departamento de Ingeniería de Sistemas e Industrial
2025

Levantamiento de requerimientos

Los requerimientos de un sistema son fundamentales para establecer las bases del desarrollo de una aplicación que cumpla con las necesidades de sus usuarios. Este documento detalla los requerimientos funcionales y no funcionales de la aplicación web destinada a conectar a los dueños de chazas en la Universidad Nacional con los estudiantes. Los requerimientos funcionales se enfocan en las características específicas y las acciones que el sistema debe realizar, como la gestión de productos y la búsqueda por parte de los clientes. Por otro lado, los requerimientos no funcionales establecen los estándares de calidad, seguridad, rendimiento y usabilidad que garantizarán una experiencia óptima para todos los usuarios. Estos elementos en conjunto aseguran el desarrollo de una solución tecnológica eficiente y accesible.

Contexto del negocio

La aplicación móvil tiene como objetivo principal mejorar la visibilidad y accesibilidad de las chazas (pequeños negocios dentro del campus) de la Universidad Nacional. Esto permitirá que los dueños de estos puestos aumenten su clientela al exponer sus productos y su ubicación de manera digital, incentivando una competencia saludable y facilitando la toma de decisiones de los estudiantes.

Por otro lado, los estudiantes obtendrán una plataforma donde pueden explorar opciones, comparar precios, productos, y ubicaciones, optimizando su experiencia dentro del campus. Esta solución fomenta la digitalización de pequeños negocios y crea un ecosistema dinámico que beneficia tanto a los vendedores como a los consumidores.

Surgimiento de la idea

La idea surge de la identificación de problemáticas y necesidades comunes entre los miembros del equipo, quienes comparten la experiencia de la vida universitaria. En particular, se busca crear un sistema de e-commerce que conecte de manera eficiente las "chazas" (establecimientos de comida informales dentro de la universidad) con los usuarios.

Esta iniciativa nace de la necesidad de facilitar el acceso a la oferta gastronómica de las chazas, superando las limitaciones de tiempo y distancia que a menudo enfrentan los estudiantes. Se espera que esta solución tecnológica mejore la experiencia de compra, tanto para los usuarios como para los propietarios de las chazas, optimizando el proceso de pedido y entrega.

Elección de la idea

La elección de la idea para el proyecto fue un proceso colaborativo en el que cada miembro del equipo aportó sus propuestas y se realizó una votación para seleccionar la opción más viable y atractiva.

Propuestas iniciales

- Aplicación de e-commerce para la universidad: Esta idea central se enfocaba en conectar a los usuarios de la universidad con los diversos establecimientos de comida y otros servicios disponibles en el campus.

- Aplicación de objetos perdidos: Se propuso una plataforma para facilitar la búsqueda y devolución de objetos perdidos en la universidad.
- Aplicación de rutinas de ejercicio: Esta idea buscaba ofrecer a los estudiantes y otros miembros de la comunidad universitaria rutinas de ejercicio personalizadas, adaptadas a las instalaciones y equipos disponibles en la universidad.

Criterios de selección

Para elegir la idea ganadora, el equipo consideró los siguientes criterios:

- Interés y motivación: Se valoró el entusiasmo de los miembros del equipo hacia cada propuesta, buscando un proyecto que motivara a todos a dar lo mejor de sí.
- Viabilidad y recursos: Se evaluó la disponibilidad de tiempo, conocimientos técnicos y recursos necesarios para llevar a cabo cada proyecto en el plazo establecido.

Resultado de la votación

Tras un análisis de las propuestas y una votación democrática, el equipo decidió enfocarse en la aplicación de e-commerce para conectar a los usuarios con las "chazas" de la universidad. Esta idea fue seleccionada por su potencial para resolver una necesidad real de la comunidad universitaria, su viabilidad técnica y el entusiasmo que generó en el equipo.

La decisión de elegir esta idea refleja el compromiso del equipo con la creación de un proyecto que no solo sea innovador, sino que también tenga un impacto positivo en la vida de los estudiantes y otros miembros de la universidad.

Problema principal

El problema principal radica en que los estudiantes de la Universidad Nacional suelen adquirir productos en las chazas del campus, pero enfrentan dificultades para identificar cuáles ofrecen los mejores precios o productos que se ajusten a sus necesidades, especialmente cuando desconocen la existencia de algunas chazas con ofertas más convenientes o específicas. Al mismo tiempo, muchas chazas tienen poca visibilidad entre los estudiantes, lo que limita su capacidad para alcanzar una mayor clientela, generando dificultades en sus ventas y, en algunos casos, reduciendo su sostenibilidad económica. Esta falta de conexión entre estudiantes y chazas representa una oportunidad para implementar una solución que beneficie a ambas partes.

¿Qué expectativas tienen los usuarios potenciales del sistema?

Expectativa de los estudiantes:

- **Ahorro de tiempo y energía**: Olvidarse de las largas caminatas y filas, y encontrar la comida perfecta en cuestión de segundos, optimizando así su valioso tiempo y energía.
- **Poder de decisión al comprar:** Acceder a la información de precios permite que los estudiantes puedan elegir la opción que mejor se adapte a sus necesidades.

- **Información al alcance de la mano**: Acceder a menús detallados, precios actualizados, descripciones de productos e incluso información nutricional, todo en un solo lugar y al alcance de un clic.

Expectativas de los dueños de chazas:

- **Visibilidad amplificada:** Aumentar el alcance de sus negocios y llegar a un público más amplio de estudiantes, dando a conocer sus productos y ofertas de manera efectiva.
- **Marketing estratégico**: Promocionar sus productos y ofertas de manera segmentada, dirigiéndose a grupos específicos de estudiantes y aumentando el impacto de sus campañas publicitarias.

¿Qué beneficios esperan ustedes al desarrollar este proyecto?

Al desarrollar este proyecto, esperamos generar un impacto positivo en la comunidad universitaria al facilitar la conexión entre estudiantes y "chazas". Buscamos mejorar la experiencia de compra y promover el desarrollo de los negocios locales, sin descartar la posibilidad de obtener beneficios económicos a largo plazo que permitan garantizar la sostenibilidad de la aplicación.

Requerimientos funcionales

Con base en esta problemática, se plantearon los siguientes requerimientos para la aplicación:

- ***Registro de vendedores***
    - Los vendedores podrán registrarse y autenticarse para gestionar el perfil de su chaza de manera segura.
- ***Adición de productos***
    - Los vendedores podrán agregar sus productos especificando su precio.
- ***Adición de fotografías del producto***
    - Se podrá adjuntar fotos del producto.
- ***Adición de códigos de barras del producto***
    - Se incluirá la funcionalidad para escanear o ingresar manualmente códigos de barras al agregar productos.
- ***Adición de ubicación de la chaza***
    - Los vendedores podrán colocar la ubicación de su chaza en el mapa del campus.
- ***Adición de fotografías de la chaza***
    - Se podrán agregar fotos de la chaza para mejorar su visibilidad y atractivo.

- ***Registro de cliente***
    - Los clientes podrán crear un perfil donde se almacenarán sus datos
- ***Inicio de sesión***
    - Los usuarios de las chazas y los clientes podrán autenticarse en el sistema para mantener sus preferencias y almacenar sus datos
- ***Búsqueda de productos por nombre***
    - Los clientes podrán buscar productos por nombre. La búsqueda mostrará las chazas que venden el producto, ordenadas por precio.
- ***Búsqueda de productos por código de barras***
    - Los clientes podrán buscar productos escaneando su código de barras. La búsqueda mostrará las chazas que venden el producto, ordenadas por precio.
- ***Gestión de chazas favoritas***
    - Los clientes podrán agregar chazas a su lista de preferidos.
- ***Gestión de Lista de Productos Favoritos***
    - Los clientes podrán agregar productos a su lista de preferidos
- ***Creación de sistema de reseñas para calificar chazas***
    - Las chazas tendrán un sistema de reseñas que permitirá a los clientes dejar comentarios sobre su experiencia.
- ***Filtro de chazas por horario de atención, categoría de productos, o medios de pago aceptados***
- ***Publicación de promociones temporales***
    - Funcionalidad para que los vendedores publiquen promociones o descuentos temporales.
- ***Notificaciones a los clientes por eventos en chazas preferidas***
    - Notificaciones para los clientes cuando una chaza preferida tenga nuevos productos, ofertas, o cambios en precios.
- ***Mapa interactivo de las chazas***
    - Implementar un mapa interactivo del campus donde los clientes puedan visualizar todas las chazas con sus ubicaciones exactas.

- ***Integración con redes sociales***
    - Posibilidad de compartir perfiles de chazas o promociones en redes sociales para aumentar su alcance.
- ***Soporte multilingüe***
    - Agregar soporte para idiomas adicionales si se identifica la necesidad dentro de la comunidad universitaria.

Requerimientos no funcionales

- ***Seguridad***
    - Proteger los datos de los usuarios mediante cifrado, especialmente para la autenticación y los medios de pago, garantizando un entorno seguro para los vendedores y clientes.
- ***Rendimiento***
    - Las búsquedas de productos y la carga de perfiles de chazas deben completarse en menos de 2 segundos para ofrecer una experiencia de usuario fluida y eficiente.
- ***Usabilidad***
    - Diseñar una interfaz intuitiva y accesible para facilitar el uso de la plataforma.

Análisis de requerimientos

Para el análisis de requerimiento se plantea clasificar inicialmente los requerimientos siguiendo el método **M**o**SC**o**W**, donde se clasifican los elementos según lo esencial que sean para la completitud del proyecto. De esta manera se tiene la siguiente convención:

- Must have (M): Requisitos esenciales para el funcionamiento del sistema.
- Should Have (S): Características importantes, pero que pueden ser pospuestas.
- Could Have (C): Deseables, pero no fundamentales para la primera versión.
- Won't Have (W): Elementos que no se incluirán en este ciclo de desarrollo.

Por ello, los requerimientos quedaron asignados de la siguiente manera:

Requerimientos funcionales

- Registro de vendedores (M)
- Adición de productos (M)
- Adición de fotografías del producto (S)
- Adición de códigos de barras del producto (C)
- Adición de ubicación de la chaza (S)
- Adición de fotografías de la chaza (C)
- Registro de cliente (M)
- Inicio de sesión (M)
- Búsqueda de productos por nombre (M)
- Búsqueda de productos por nombre por código de barras (C)
- Gestión de chazas favoritas (S)
- Gestión de Lista de Productos Favoritos (S)
- Creación de sistema de reseñas para calificar chazas (C)
- Filtro de chazas por horario de atención, categoría de productos o medios de pago aceptados (C)
- Publicación de promociones temporales (W)
- Notificaciones a los clientes por eventos en chazas preferidas (W)
- Mapa interactivo de las chazas (W)
- Integración con redes sociales (W)
- Soporte multilingüe (W)

Requerimientos no funcionales

- Seguridad (M)
- Rendimiento (S)
- Usabilidad (S)

Una vez realizada esta clasificación, solamente se continuará el proceso con las historias de usuario correspondientes a las prioridades Must, Should y Could, siguiendo las indicaciones proporcionadas. Las historias clasificadas como Won't se descartan en esta etapa, ya que no representan una prioridad actual y su desarrollo implicaría un desgaste innecesario de recursos. Además, este enfoque permite adaptarnos de manera más eficiente a los cambios de prioridades en ciclos futuros, cumpliendo con la metodología iterativa y evolutiva de gestión de requerimientos.

Posteriormente, se desarrolla el proceso de estimación de tiempos basada en la sucesión de Fibonacci, donde cada requerimiento requerirá un número de días perteneciente a esta secuencia. Además, se organiza en un orden lógico de secuencia, con posibilidad de ser modificado en el futuro.

Must

- Registro de vendedores: 2 días, debido a que implica configurar el sistema de autenticación y establecer una base de datos para almacenar los perfiles de los vendedores de forma segura.
- Registro de cliente: 2 días, considerando la necesidad de desarrollar una funcionalidad similar al registro de vendedores, pero adaptada al perfil de cliente.
- Inicio de sesión: 1 día, considerando que se debe crear la pantalla de inicio de sesión para autenticación, además de realizar las configuraciones necesarias para poder utilizar la base de datos para poder comprobar la identidad del usuario.
- Adición de productos: 2 días, ya que requiere diseñar la interfaz de usuario y vincularla con la base de datos para registrar los productos de manera eficiente.
- Búsqueda de productos por nombre: 2 días, debido a que se requiere implementar un sistema de búsqueda optimizado.
- Seguridad: 3 días, ya que incluye la implementación de cifrado, manejo seguro de contraseñas, y pruebas exhaustivas para proteger los datos de los usuarios.

Should

- Usabilidad: 3 días, para diseñar y probar una interfaz amigable.
- Adición de fotografías del producto: 3 días, debido a que implica diseñar una funcionalidad para cargar, almacenar y mostrar imágenes en el perfil de los productos.
- Adición de ubicación de la chaza: 3 días, considerando la integración con servicios de mapas y la representación de la ubicación dentro del campus.
- Gestión de chazas favoritas: 1 día, dado que es necesario implementar una relación entre los perfiles de los clientes y las chazas en la base de datos.
- Gestión de Lista de Productos Favoritos: 1 día, similar al caso anterior, pero centrado en los productos.
- Rendimiento: 3 días, ya que se necesita optimizar las consultas y la arquitectura del sistema para asegurar tiempos de respuesta adecuados.

Can

- Adición de códigos de barras del producto: 2 días, considerando el diseño de un sistema que permita registrar y consultar productos a través de códigos de barras.
- Búsqueda de productos por código de barras: 2 días, debido a que se requiere implementar un sistema de escaneo de código de barras del producto y realizar la conexión a la base de datos de productos
- Adición de fotografías de la chaza: 3 días, debido a que requiere una funcionalidad similar a la de las fotografías de productos, pero adaptada al perfil de las chazas.
- Creación de sistema de reseñas para calificar chazas: 3 días, ya que implica diseñar la funcionalidad para capturar, almacenar y mostrar reseñas, además de permitir la moderación.

- Filtro de chazas por horario de atención, categoría de productos o medios de pago aceptados: 3 días, considerando la implementación de filtros dinámicos y su integración con la base de datos.

Con base en este planteamiento se puede crear el siguiente cronograma prototipo para empezar el desarrollo:

Estimación de requerimientos		
Prioridad	**Requerimiento**	**Estimación**
Must	Registro de vendedores	2
	Registro de clientes	2
	Inicio de sesión	1
	Adición de productos	2
	Búsqueda de productos por nombre	2
	Seguridad	3
	Subtotal Acumulado	**12**
Should	Usabilidad	3
	Adición de fotografías del producto	3
	Adición de ubicación de la chaza	3
	Gestión de chazas favoritas	1
	Gestión de Lista de Productos Favoritos	1
	Rendimiento	3
	Subtotal Acumulado	**26**
Can	Adición de códigos de barras del producto	2
	Búsqueda de productos por código de barras	2
	Adición de fotografías de la chaza	3
	Sistema de reseñas para calificar chazas	3
	Filtro de chazas por horario de atención, categoría de productos o medios de pago aceptados	3
Total Estimación		**39**

De esta manera, los requerimientos que tienen que estar (M), se desarrollarán en 12 días. Al incluir los requerimientos que deberían estar (S), se llega a un total de 26 días y si se incluye el requerimiento que puede estar (C), se llega a un total estimado de duración del desarrollo del proyecto de 39 días.

Sumado a este análisis se puede plantear si algunos de los requerimientos se encuentran en conflicto, sin embargo, en el alcance actual no se observa ningún tipo de conflicto.

Análisis de gestión de software

En la gestión de proyectos de software, es fundamental analizar la relación entre tiempo, costo y alcance para garantizar una planificación efectiva. Estos tres factores están interconectados, y cualquier modificación en uno de ellos impacta directamente en los demás. Este análisis permitirá estimar el tiempo requerido para el desarrollo de cada módulo, calcular los costos asociados considerando sueldos, licencias, infraestructura y herramientas externas, y definir con claridad los límites del sistema dentro del MVP. Además, se investigará el valor real del trabajo en el mercado para establecer expectativas alineadas con la industria y evitar subvaloraciones en la estimación de costos.

Alcance

El alcance del proyecto se definirá con base en la priorización de requerimientos mediante el método MoSCoW. Para el MVP (Producto Mínimo Viable), se incluirán los requerimientos clasificados como "Must have" y "Should have", es decir, tanto los elementos esenciales para el funcionamiento del sistema como aquellos que, aunque no sean críticos, aportan un valor significativo a la experiencia del usuario y mejoran la funcionalidad del producto.

Esta decisión se fundamenta en la necesidad de ofrecer una solución viable y competitiva desde la primera versión, asegurando que el sistema no solo cumpla con su propósito principal, sino que también brinde una experiencia de uso más completa. Al incluir los requerimientos "Should have", se reduce la necesidad de actualizaciones inmediatas tras el lanzamiento, lo que permite enfocar recursos en la optimización y validación del producto antes de considerar futuras expansiones.

Tiempo

El tiempo estimado para el desarrollo del proyecto se ha determinado a partir del análisis de requerimientos utilizando la metodología MoSCoW. Según esta evaluación, el desarrollo de los requerimientos clasificados como "Must have" requiere un total de 12 días, mientras que los correspondientes a "Should have" suman 14 días adicionales. En consecuencia, el tiempo total estimado para completar el MVP asciende a 26 días.

Esta planificación considera las etapas necesarias para cada funcionalidad, incluyendo diseño, desarrollo y pruebas, asegurando que el producto final sea funcional y estable antes de su lanzamiento.

Costo

Para el desarrollo del proyecto, se contará con un equipo conformado por un desarrollador Flutter, encargado de la implementación del frontend móvil, y dos desarrolladores backend, uno senior y otro junior, responsables de la construcción y mantenimiento de la API REST en NestJS, así como de la gestión de la base de datos. Además, se incluirá un diseñador UX/UI, quien se enfocará en la creación de interfaces intuitivas y una experiencia de usuario optimizada. La coordinación y planificación del proyecto estarán a cargo de un Project Manager, asegurando una adecuada gestión de tiempos, recursos y comunicación entre los miembros del equipo.

Para estimar el presupuesto del equipo de desarrollo, se consideran los siguientes roles y sus remuneraciones promedio en el mercado laboral a lo largo de un mes, teniendo en cuenta que el tiempo de desarrollo del proyecto son 26 días laborales:

- Desarrollador Flutter
- Desarrollador Backend NestJS Senior
- Desarrollador Backend NestJS Junior
- Diseñador UX/UI
- Project Manager

Por otra parte, para el despliegue en tiendas móviles, se incurrirá en un costo único de $25 USD para la creación de la cuenta de desarrollador en Google Play Store y $99 USD anuales para la cuenta en App Store. En cuanto a la infraestructura en la nube, se utilizará Render para el despliegue del backend y la base de datos PostgreSQL. Render ofrece un plan gratuito para pruebas, pero se estima que el costo mensual de un plan de pago para producción (con 1 GB de RAM para el backend y 10 GB de almacenamiento en la base de datos) será de $17 USD/mes. Adicionalmente, se utilizará la API de Google Maps, la cual tiene un crédito gratuito mensual de $200 USD. Este crédito es suficiente para cubrir la mayoría de las operaciones, como geolocalización y visualización de mapas, siempre que el tráfico se mantenga dentro de los límites de uso gratuito.

El desarrollo de la aplicación se basará en tecnologías open-source: Flutter para el frontend y NestJS para el backend, lo que no implica costos adicionales por las herramientas de desarrollo.

En total, los costos del proyecto quedan detallados de la siguiente manera:

Categoría	Descripción	Estimación	Costo promedio (USD)	Costo promedio (COP)
Trabajadores	Desarrollador Flutter	2.800.000 COP a 4.800.000 COP mensuales.	926,83	3.800.000
	Desarrollador Backend NestJS Senior	6.000.000 COP a 10.000.000 COP mensuales.	1951,22	8.000.000
	Desarrollador Backend NestJS Junior	2.500.000 COP a 4.500.000 COP mensuales.	853,66	3.500.000
	Diseñador UX/UI	2.500.000 COP a 4.500.000 COP mensuales.	853,66	3.500.000
	Project Manager	6.000.000 COP a 10.000.000 COP mensuales.	1951,22	8.000.000
Licencias	Cuenta de desarrollador Google Play	25 USD pago único	25,00	102.500
	Cuenta de desarrollador AppStore (anual)	99 USD	99,00	405.900
Servicios en la nube	Render (mensual)	17 USD	17,00	69.700
Herramientas	API Google Maps	0 USD	0,00	0

externas				
TOTAL			6677,59	27.378.100

Las estimaciones de los salarios fueron mayormente obtenidas al realizar investigación en plataformas como GlassDoor, Indeed Colombia, Computrabajo y LinkedIn, aunque en estos últimos muchas veces en las ofertas de trabajo no se muestra el salario o rango de salario esperado.

Diseño y Arquitectura

El diseño y la arquitectura de un sistema son fundamentales para garantizar su eficiencia, escalabilidad y mantenimiento a lo largo del tiempo. Una planificación adecuada permite estructurar el flujo de datos, optimizar el rendimiento y facilitar futuras modificaciones sin afectar la estabilidad del proyecto.

En este apartado se presentan dos secciones clave: Arquitectura del Sistema, donde se describe la organización de los componentes y su interacción, y Diseño de Base de Datos, donde se detalla la estructura de almacenamiento, las relaciones entre entidades y la justificación del modelo elegido. Ambos aspectos son esenciales para asegurar la solidez y el correcto funcionamiento de la aplicación.

Arquitectura del Sistema

El proyecto sigue una arquitectura cliente-servidor, donde el frontend en Flutter se comunica con el backend en NestJS a través de una API REST. El backend, a su vez, gestiona las solicitudes de datos, interactuando con la base de datos PostgreSQL para el almacenamiento de información y con la API de Google Maps para obtener datos de ubicación y mapas.

Esta arquitectura fue elegida por su simplicidad y eficiencia, permitiendo una clara separación de responsabilidades entre la interfaz de usuario y la lógica de negocio. Al utilizar Flutter, se garantiza una experiencia multiplataforma eficiente para dispositivos móviles, mientras que NestJS, basado en Node.js, proporciona un backend escalable y modular. PostgreSQL se seleccionó por su capacidad para manejar consultas complejas y su estabilidad en aplicaciones de producción.

La estructura cliente-servidor permite futuras mejoras, como la implementación de autenticación, optimización del rendimiento mediante almacenamiento en caché o la posible incorporación de nuevos servicios sin afectar la base del sistema. Esta arquitectura es adecuada para los requisitos del proyecto y se adapta bien a los recursos disponibles.

Base de datos

El diseño de la base de datos se centra en los casos de uso iniciales de la aplicación, priorizando la gestión de chazas, productos y usuarios. Se implementará un CRUD (Crear, Leer, Actualizar, Eliminar) para la gestión de chazas, permitiendo a los usuarios crear sus propias chazas. Cada chaza podrá tener múltiples productos y categorías, facilitando futuras implementaciones.

Estructura de la Base de Datos

La base de datos se compone de las siguientes tablas, cada una con un propósito específico:

- **Usuarios**: Almacena información de los usuarios, utilizando UUID como clave primaria por seguridad.
- **Chazas**: Almacena información de las chazas, incluyendo su ubicación, descripción y relación con el usuario que la creó.
- **Productos**: Almacena información de los productos, incluyendo su categoría y código de barras.
- **Categorias**: Almacena información de las categorías de productos.

- **Photo**: Almacena las URLs de las fotos de las chazas y productos, las cuales se guardarán en un bucket (Firebase Storage o AWS S3).
- **Chaza_Producto**: Tabla de relación muchos a muchos entre Chazas y Productos, incluyendo el precio y la cantidad.
- **Chazas_favoritas**: Almacena las chazas favoritas de cada usuario.
- **Productos_favoritos**: Almacena los productos favoritos de cada usuario.
- **Calificacion**: Almacena las calificaciones y comentarios de los usuarios sobre las chazas.
- **Medios_pago**: Almacena los medios de pago aceptados por cada chaza.
- **Horario_chaza**: Almacena los horarios de atención de cada chaza.

Relaciones entre Tablas

Las tablas se relacionan de la siguiente manera:

- **Uno a Muchos**: Un usuario puede tener muchas chazas, una chaza puede tener muchos productos, una categoría puede tener muchos productos, etc.
- **Muchos a Muchos**: Una chaza puede tener muchos productos y un producto puede pertenecer a muchas chazas (a través de la tabla Chaza_Producto).

Claves Primarias

Se utilizan IDs enteros (INT) como claves primarias para todas las tablas, excepto la tabla Usuarios que utiliza UUID por seguridad. Se eligió INT en lugar de DOUBLE para las claves primarias debido al crecimiento esperado de usuarios y para optimizar el uso de memoria.

Almacenamiento de Fotos

Las URLs de las fotos se almacenarán en la tabla Photo, mientras que los archivos de imagen se guardarán en un bucket (Firebase Storage o AWS S3). La elección de la plataforma dependerá de la implementación final.

Índices y Triggers

Debido a la alta frecuencia de consultas relacionadas con las tablas "chazas" y "usuarios", se decidió crear índices en sus claves primarias, "id_chaza" e "id_usuario", con el objetivo de optimizar las búsquedas en la base de datos, adicionalmente los triggers por ahora en esta fase no se ven necesarios pero no se exceptúa su implementacion durante el desarrollo .