

# Instituto Politécnico Nacional

## Escuela Superior de Cómputo

DESARROLLO DE APLICACIONES MOVILES NATIVAS.

Proyecto Final

*Profesor: Vélez Saldaña Ulises*

*Alumnos:*

*Almazán Martínez Angel Jesús*

*Rolón Cárdenas Roberto*

*Trejo García Lizette*

*GRUPO*

*7CM3*

*16 de Enero de 2024*

# Contents

<b>1</b>	<b>Objetivo</b>	<b>3</b>
<b>2</b>	<b>Alcance del proyecto</b>	<b>3</b>
<b>3</b>	<b>Introducción</b>	<b>3</b>
<b>4</b>	<b>Desarrollo</b>	<b>4</b>
4.1	Etapa de Análisis y Diseño . . . . .	4
4.1.1	User research . . . . .	4
4.1.2	Affinity Map: Hallazgos y deducciones significativas . . . . .	5
4.1.3	User Personas . . . . .	6
4.2	Etapa de Desarrollo . . . . .	11
4.2.1	Desarrollo de la aplicación diseñada con Kotlin y JetPack Compose . . . . .	11
4.2.2	Diseño de los endpoints el REST de los servicios web para acceder al backend . . . . .	23
4.2.3	Diseño de la base de datos en su tercera forma normal . . . . .	23
4.2.4	Desarrollo del Backend . . . . .	24
<b>5</b>	<b>Conclusión</b>	<b>28</b>
<b>6</b>	<b>Referencias Bibliográficas</b>	<b>28</b>

## List of Figures

1	Mapa de Afinidad . . . . .	5
2	User Person 1 . . . . .	6
3	User Person 2 . . . . .	7
4	User Person 3 . . . . .	8
5	User Person 4 . . . . .	9
6	User Person 5 . . . . .	10
7	Pantalla Principal . . . . .	11
8	Creación de cuenta . . . . .	11
9	Inicio de Sesión . . . . .	12
10	Alerta de campos . . . . .	12
11	Catálogo y Categorías . . . . .	13
12	Carrito con productos . . . . .	13
13	Pago del Carrito . . . . .	14
14	Creación de un pedido . . . . .	14
15	Detalles de Pedido . . . . .	15
16	Pago del Carrito . . . . .	16
17	Creación de un pedido . . . . .	16
18	Usuarios vista del Admin . . . . .	17
19	Agregar Usuario . . . . .	17
20	Productos a la venta . . . . .	18
21	Editar Producto . . . . .	18
22	Eliminar Producto . . . . .	19
23	Agregar Producto . . . . .	19
24	Vista Categorías . . . . .	20
25	Editar Producto . . . . .	20
26	Eliminar Categoría . . . . .	21
27	Agregar Categoría . . . . .	21
28	Asignar Repartidor . . . . .	22
29	Visualizar pedidos realizados . . . . .	22
30	Visualizar pedidos realizados . . . . .	23
31	Label e IconButton . . . . .	24
32	Card para productos . . . . .	24
33	Sealed Class para los productos . . . . .	25
34	Objects para la barra de navegación . . . . .	25
35	Enum class para la barra de navegación . . . . .	25
36	Card para ubicaciones . . . . .	26
37	FAB para ubicaciones . . . . .	26
38	LifecycleOwner para ubicaciones . . . . .	27
39	Corrutina para los Usuarios del Admin . . . . .	27
40	Llamadas a API . . . . .	27
41	Object de la URL de la API . . . . .	28

# 1 Objetivo

Construir una aplicación Móvil para pedidos de una Food-Truck para que puedan hacerse pedidos en infraestructuras como las de Zacatenco.

## 2 Alcance del proyecto

De manera general se indican a continuación los principales puntos a considerar:

- El sistema está pensado para negocios de comida conocidos como Food-Truck, los cuales se instalan cerca de zonas pobladas, específicamente: Escuelas, campus como Zacatenco, edificios gubernamentales, etc.
- El sistema debe considerar pedidos y entrega a pie y en bicicleta en una cierta zona predefinida.
- El sistema debe considerar pago con tarjeta, previo a la entrega y la posibilidad de pagar al recibir el paquete.

## 3 Introducción

En un mundo donde la industria de los Food-Trucks ha florecido como una alternativa fresca y deliciosa para satisfacer el apetito de aquellos que buscan comidas rápidas y deliciosas, estos vehículos rodantes se han convertido en un pilar de la oferta gastronómica en zonas pobladas, incluyendo escuelas, campus universitarios, edificios gubernamentales y otros lugares concurridos.

Se propone la construcción de una aplicación móvil que permita realizar pedidos a Food-Trucks con una atención especial en infraestructuras como las de Zacatenco. Esta aplicación no solo revolucionará la forma en que los clientes disfrutan de sus comidas favoritas, sino que también mejorará la eficiencia y comodidad para los propietarios de los Food-Trucks.

A lo largo de este proceso, se abordarán aspectos esenciales de la gestión de pedidos, seguridad, eficiencia y satisfacción del cliente. La aplicación móvil servirá como una plataforma de pedidos, tal como un puente que conecta a los Food-Trucks con los clientes.

Este documento explora los desafíos y oportunidades que ofrece este emocionante proyecto, detallando aspectos clave como la interfaz de usuario, la gestión de pedidos, así como la importancia de la seguridad y confiabilidad en las transacciones. A medida que avanzamos en la construcción de esta aplicación, nos enfocamos en mejorar la experiencia del cliente.

## 4 Desarrollo

### 4.1 Etapa de Análisis y Diseño

#### 4.1.1 User research

- *Objetivo del User research*

Obtener información clave sobre las preferencias y expectativas de los usuarios finales, incluyendo estudiantes de Zacatenco, trabajadores de edificios gubernamentales y otros posibles consumidores de Food-Truck, con el propósito de diseñar una aplicación móvil altamente funcional y centrada en el usuario que permita la realización de pedidos de manera eficiente, la elección de opciones de pago, y una entrega conveniente y satisfactoria en la zona predefinida.

- *Técnicas utilizadas*

Para este proyecto se utilizó:

1. Encuesta
2. Presentación de datos de la encuesta
3. Análisis de la competencia

- *Organización de la información resultante*

En el repositorio de GitHub se encuentran 3 archivos, los cuáles contienen lo siguiente en cada uno:

1. *Análisis de competencia:* Se evalúa la competencia de las aplicaciones Uber, Didi y Rappi en las siguientes áreas:
  - Características
  - Usabilidad
  - Precio
  - Marketing
  - Metodología
2. *Encuesta:* Se encuentran los resultados de la encuesta realizada a alumnos de entre 18 y 22 años de edad, que estudian en Zacatenco.
3. *Presentación resultados encuesta:* Contiene información para tomar en cuenta en el desarrollo de una aplicación móvil destinada a facilitar los pedidos en Food-Trucks en zonas como Zacatenco. Conociendo las preferencias, necesidades y expectativas. Estos datos nos ayudarán a diseñar una aplicación que satisfaga de manera óptima las demandas de nuestros usuarios y mejore su experiencia al ordenar comida de Food-Truck.

- *Liga al repositorio*

- <https://github.com/rrolon13/Android-Project/tree/main/user%20search>

#### 4.1.2 Affinity Map: Hallazgos y deducciones significativas

Para completar este punto se hizo el uso de un mapa de afinidad:

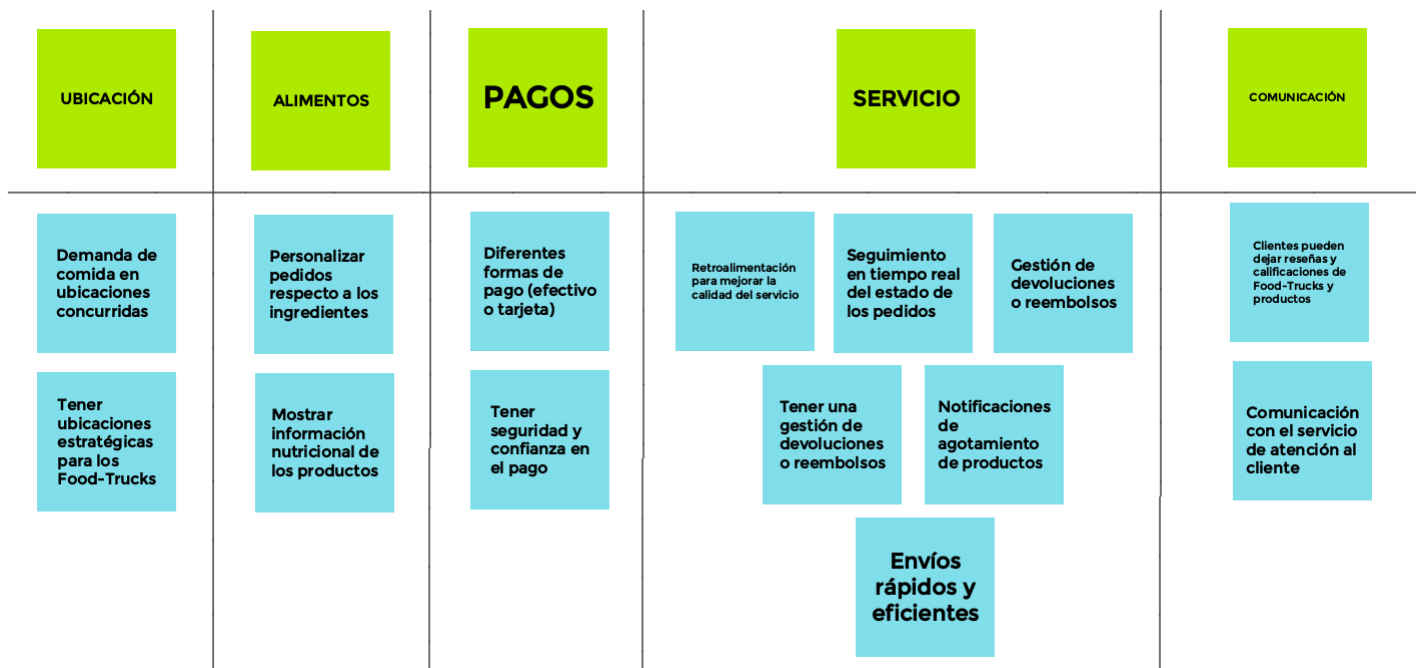


Figure 1: Mapa de Afinidad

Para identificar lo siguiente:

- **Lista de Hallazgos**

- La disponibilidad de opciones de pago, tanto con tarjeta como en efectivo, demuestra la necesidad de adaptarse a las preferencias de pago de los clientes.
- La combinación de ubicación estratégica, métodos de entrega y opciones de pago apunta a la importancia de brindar una experiencia positiva al usuario.

- **Lista de Deducciones significativas**

- Los Food-Trucks se instalan en lugares donde hay una alta afluencia de personas, lo que indica una demanda de comida conveniente en esas ubicaciones.
- La inclusión de métodos de entrega a pie y en bicicleta sugiere la importancia de una entrega rápida y eficiente para atender a los clientes en zonas cercanas.
- Con las opciones de pago, especialmente el pago con tarjeta antes de la entrega, se debe garantizar la seguridad y la confianza del cliente en el sistema.

### 4.1.3 User Personas

- **USER PERSON 1**

	<h2>Roberto</h2> <p><b>Estudiante de la zona de Zacatenco</b></p> <p>Edad: entre 18 a 25 años Ocupación: Estudiante Ubicación: Escuela Superior de Cómputo</p>	<p>Su amigo, Joaquín, lo llevó a probar la comida de los food trucks de Zacatenco; a Roberto le gustó mucho la comida ofrecida por estos establecimientos.</p>
<p><b>FRUSTRACIONES</b></p> <ul style="list-style-type: none"><li>• Le es difícil desplazarse en los 30 minutos de receso que tiene hasta la zona donde estos se encuentran para consumir de los mismos.</li><li>• Lamenta haber probado algo tan rico pero que le quede tan fuera de su alcance ya que ni él ni sus amigos cuentan con vehículo propio para visitar la zona ya mencionada</li></ul>	<p><b>CONOCIMIENTOS</b></p> <p>Roberto sabe de la existencia de aplicaciones como UberEats, Rappi y Didifood, pero ninguna de estas contempla estos foodtrucks como parte de los establecimientos desde los que se les puede enviar tan deliciosos manjares.</p>	<p><b>A Roberto le gustaría poder tener una aplicación en la que pueda pedir de estos establecimientos y que le permitiera pagar con tarjeta porque a Roberto no le gusta usar efectivo</b></p>

Figure 2: User Person 1

- *USER PERSON 2*

## Ernesto

**Dueño de uno de los food-trucks en la zona de Zacatenco**

- Ubicación: Zacatenco
- Ocupación: Dueño de un food-truck

Ernesto se siente abrumado ya que de no lograr aumentar sus ventas, va a tener que retirar su camioneta de comida de la zona pues no genera las suficientes ganancias para mantener el mismo, por lo que él se acerca a los estudiantes para conocer que soluciones le puede dar a los mismos con la finalidad de aumentar sus ventas.

### PROBLEMAS

- Las ventas no van tan bien como cuando inauguró
- Las largas distancias y tiempos de traslado desde las unidades académicas de cada alumno hasta la zona de food trucks.

### SOLUCIÓN

**Usar un sistema de reparto hasta la escuela.** Ernesto está considerando registrarse en las aplicaciones de tipo delivery como UberEats, DidiFood, Rappi, etc, pero se ha encontrado con las cuotas que cobran estas aplicaciones, lo que haría que suba sus precios, resultado que, según la opinión de los mismos alumnos, desalentaría a su clientela, ya que justamente buscan consumir del local de Ernesto por sus precios accesibles para la comunidad estudiantil.



Figure 3: User Person 2



- **USER PERSON 3**


	<h2>Axel</h2> <p><b>Estudiante de bajos recursos de la ESCA Zacatenco</b></p> <p>Edad: entre 18 y 23 años Ocupación: Estudiante Ubicación: ESCA Zacatenco</p>	<p>Axel no tiene una forma rápida de generar ingresos que interfiera lo mínimo posible con su horario y carga académica, su situación económica le ha impedido visitar la nueva zona de Foodtruck que se ha inaugurado cerca de su escuela.</p>
	<p><b>MOTIVACIONES</b></p> <p>A Axel le han dicho todos sus amigos y conocidos que debería probar la comida de estos camiones</p> <p>En su regreso a casa, Axel oye como el dueño de uno de los camiones está platicando con un alumno al que le pregunta la razón de por qué después de la inauguración no ha vuelto, lo que le permite a Axel darse cuenta de la necesidad de Ernesto por distribuir su comida de forma rápida a lo largo de la zona.</p> <p><b>SOLUCIÓN</b></p> <p><b>Axel está emocionado pues sabe que podría ofrecerle a Ernesto distribuir su comida a cambio de un poco de la misma o de una buena propina</b>, sin embargo, no le queda claro del todo como en caso de hacerlo, Axel podría asegurarse de que al alumno al que le entregue el pedido, sea el que realmente lo haya pedido y no un impostor.</p>	<p><b>FRUSTRACIONES</b></p> <p>Axel se siente extremadamente triste, pues no tiene forma de probar lo que todos sus compañeros le dicen.</p>

Figure 4: User Person 3

• **USER PERSON 4**

## Marco

### Estudiante de la zona de Zacatenco

- Edad: entre 18 y 25 años
- Ubicación: Escuela Superior de Cómputo
- Ocupación: Estudiante

#### ACERCA DEL USUARIO

A Marco no le gusta la comida de su cafetería pero tampoco le gusta salir de su escuela, por ello, Marco utiliza constantemente aplicaciones de delivery como UberEats, Rappi y Didifood.

#### PROBLEMA

Los malvados restaurantes y establecimientos cercanos a su escuela han eliminado el efectivo como método de pago, razón por la que Marco ha estallado en ira y quiere reemplazar las tres aplicaciones que regularmente usa por una que obligue a los establecimientos a aceptar el efectivo como forma de pago.

Marco está pensando en pagarle a uno de sus conocidos para que vaya por comida a la plaza que tiene cerca de su escuela, pero preferiría no hacerlo, le gustan más las aplicaciones

#### NECESIDAD

Marco no está interesado en abrir una cuenta de banco, así que solo aceptará una aplicación de delivery en la que si o si se acepte el pago en efectivo.

Cuando escuchó sobre la inauguración de una zona food truck en zacatenco, se emocionó por la posibilidad de que esta nueva zona sea justo lo que **satisfaga sus necesidades de pedir comida**.



Figure 5: User Person 4

- **USER PERSON 5**

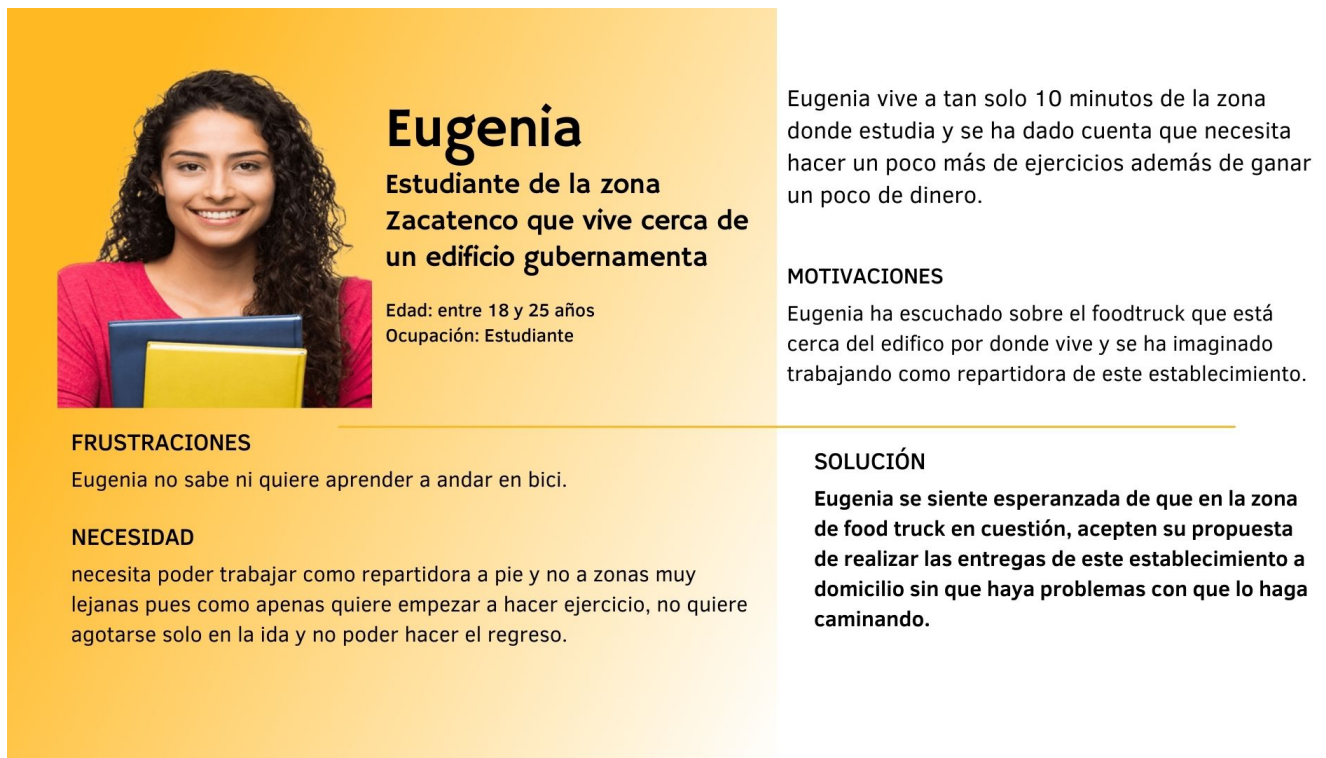


Figure 6: User Person 5

## 4.2 Etapa de Desarrollo

### 4.2.1 Desarrollo de la aplicación diseñada con Kotlin y JetPack Compose

La aplicación como se menciona, se desarrolló con ayuda de Kotlin y JetPackCompose. A continuación se muestran las interfaces de la aplicación y como es que funciona:

- Comenzamos con la primera pantalla, que es la bienvenida:



Figure 7: Pantalla Principal

- Para entrar a la aplicación primero debemos crear una cuenta en el caso de que no tengamos una:



Figure 8: Creación de cuenta

- Después para el inicio de sesión muestra lo siguiente:

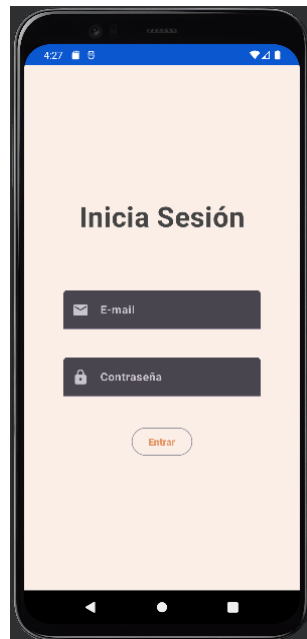


Figure 9: Inicio de Sesión

- En ambos casos, del inicio de sesión y la creación de cuenta, si no escribimos nada en las casillas, nos mostrará la siguiente alerta:

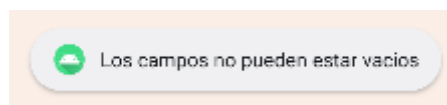


Figure 10: Alerta de campos

## VISTA USUARIO

- Una vez ingresando podemos observar el menú con las categorías que existen:

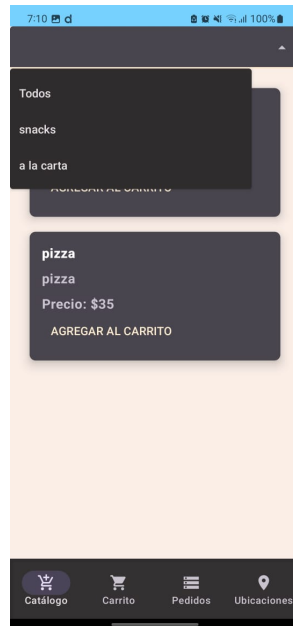


Figure 11: Catálogo y Categorías

- Al dar click en "Agregar al carrito" los productos se guardan y en la sección de "Carrito" puedes ver los productos guardados:

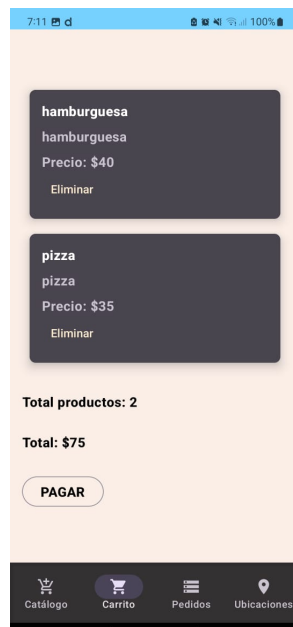


Figure 12: Carrito con productos

- Al dar click en el botón de Pagar nos llevará a la ventana del pago:

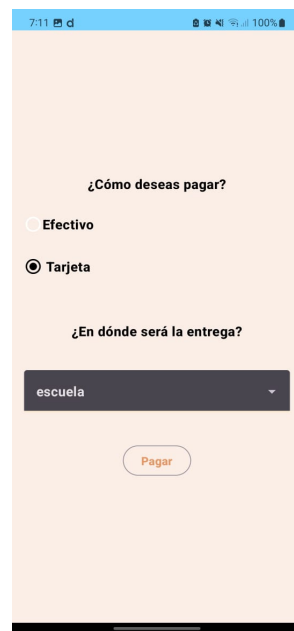


Figure 13: Pago del Carrito

- En la ventana de Pedidos tenemos nuestra orden ya creada mostrando los datos de la compra así como el status del pedido:

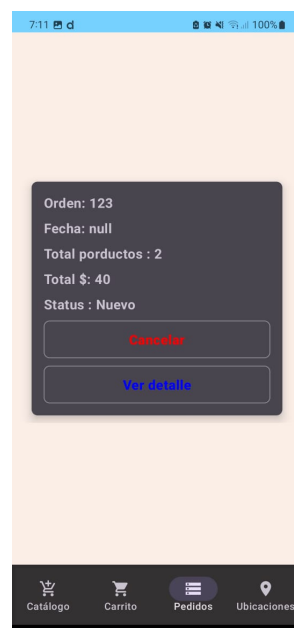


Figure 14: Creación de un pedido

- Al dar click en el botón de Ver Detalles nos parecen los detalles del pedido que realizamos:

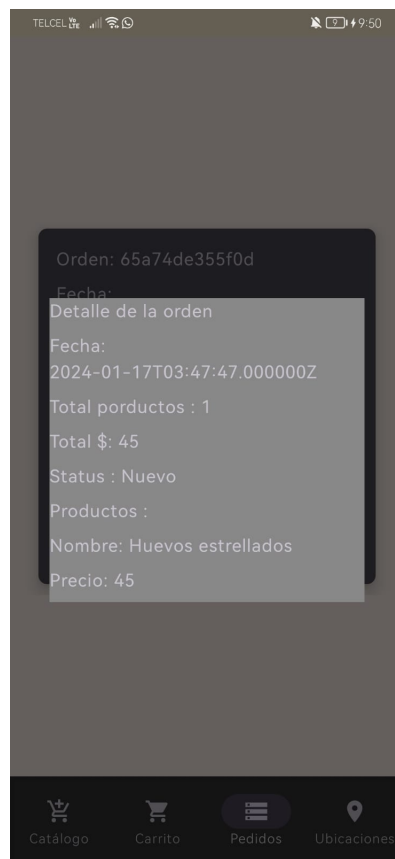


Figure 15: Detalles de Pedido



- En la ventana de Ubicaciones podemos observar las ubicaciones que tenemos guardadas para las direcciones de los pedidos:



Figure 16: Pago del Carrito

- Si deseamos agregar una ubicación sólo debemos dar click en en botón + y llenar los datos de la ubicación, después damos click en agregar y la ubicación se guardará:



Figure 17: Creación de un pedido

## VISTA ADMINISTRADOR

Para ingresar a la vista de administrador tenemos una cuenta asignada:

**Usuario:** admin@admin

**Contraseña:** 12345678

- En la ventana de Usuarios podemos ver al administrador y a los repartidores que tengamos:

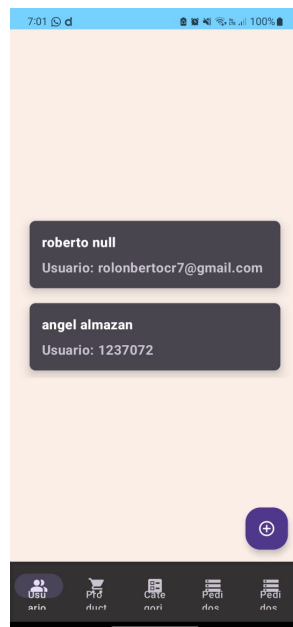


Figure 18: Usuarios vista del Admin

- Si deseamos agregar un usuario sólo debemos dar click en en botón + y llenar los datos del usuario, después damos click en crear y obtendremos un nuevo usuario:

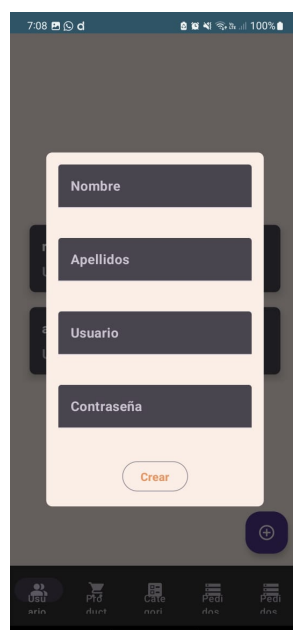


Figure 19: Agregar Usuario

- En la ventana de Productos podemos ver los productos que tengamos a la venta:

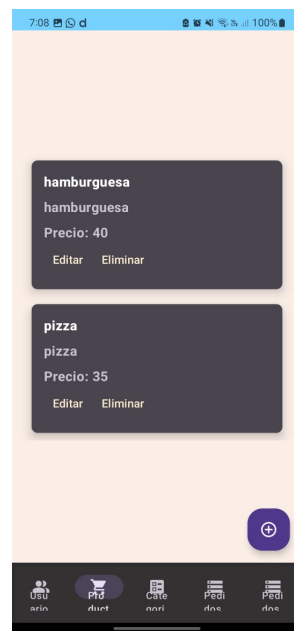


Figure 20: Productos a la venta

- Si deseamos editar el producto solo debemos dar click en el botón de Editar y hacemos los cambios necesarios:

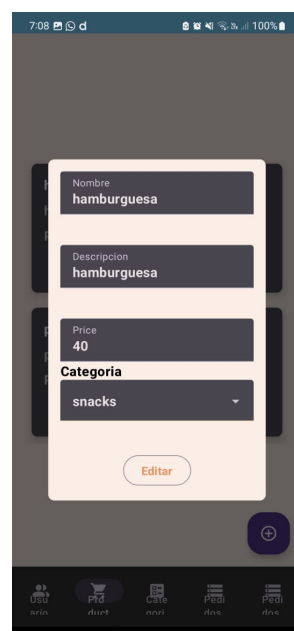


Figure 21: Editar Producto

- Si deseamos eliminar el producto solo debemos dar click en el botón de Eliminar y el producto se desecha:

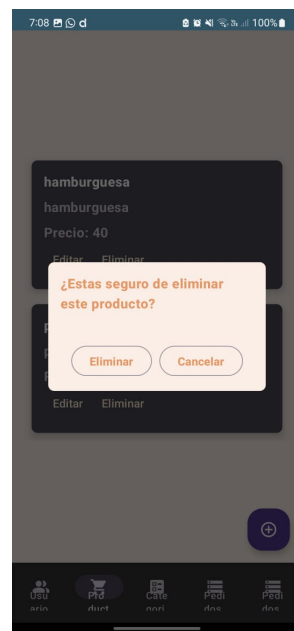


Figure 22: Eliminar Producto

- Si queremos agregar un producto solo debemos dar click en el botón + y llenamos los datos necesarios para agregarlo:

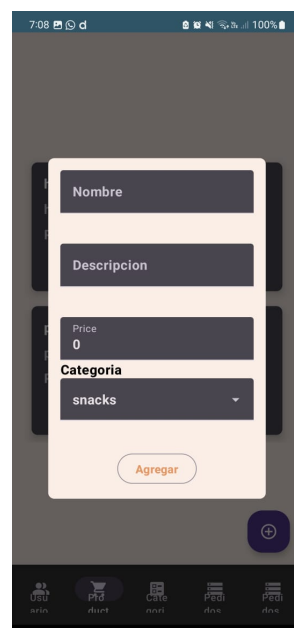


Figure 23: Agregar Producto

- Al dar click en Categorías podemos ver las categorías que existen en el sistema dependiendo de los productos que hayan:

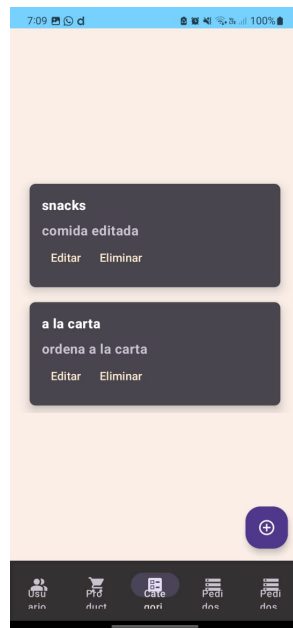


Figure 24: Vista Categorías

- Si queremos editar una categoría solo debemos dar click en el botón Editar y hacemos los cambios necesarios:

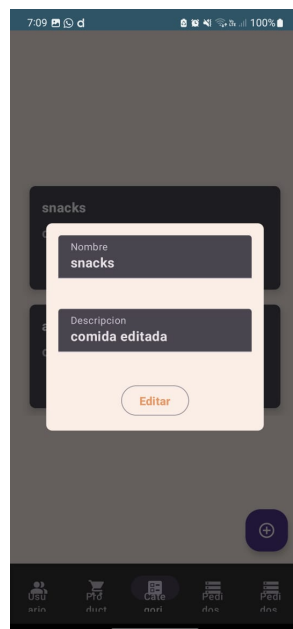


Figure 25: Editar Producto

- Si deseamos eliminar la categoría solo debemos dar click en el botón de Eliminar y la categoría se desecha:



Figure 26: Eliminar Categoría

- Si queremos agregar una categoría solo debemos dar click en el botón + y llenamos los datos necesarios para agregarla:

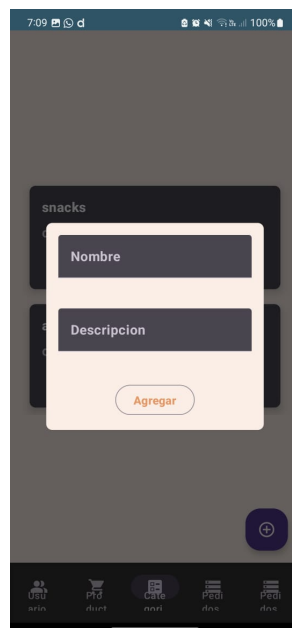


Figure 27: Agregar Categoría

- En la ventana de los Pedidos Asignados podemos ver todos los pedidos a los que se deban asignar un repartidor:

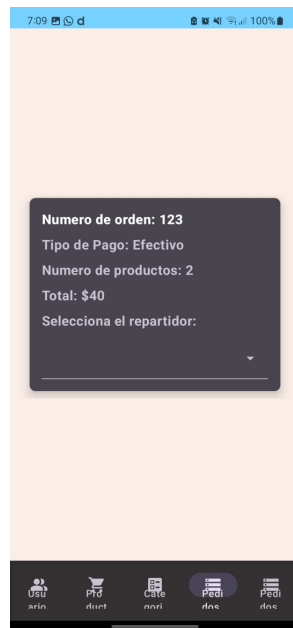


Figure 28: Asignar Repartidor

- En la ventana de los Pedidos podemos observar los pedidos que se han realizado:

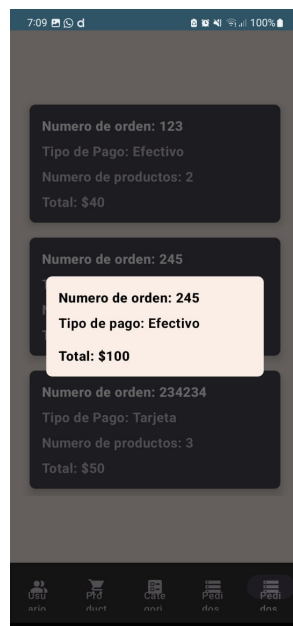


Figure 29: Visualizar pedidos realizados

#### 4.2.2 Diseño de los endpoints el REST de los servicios web para acceder al backend

Para esta sección ingresamos una url direccionando al link de los endpoints:

<https://moviles.rmo.industries/api/documentation>

#### 4.2.3 Diseño de la base de datos en su tercera forma normal

Para la base de datos se creó el siguiente diagrama:



Figure 30: Visualizar pedidos realizados



#### 4.2.4 Desarrollo del Backend

A continuación se muestran fragmentos de código que se ocuparon para la creación de la aplicación:

- Para la creación de las casillas para escribir ocupamos **Label** e **IconButton** para un mejor diseño de cada casilla:

```
label = {
  Text(text = "E-mail")
},
leadingIcon = {
  IconButton(onClick = { /*TODO*/ }) {
    Icon(
      imageVector = Icons.Filled.Email,
      contentDescription = "Email icon"
    )
  }
},
```

Figure 31: Label e IconButton

- Para mostrar los productos hicimos uso de **Card**, para mostrar los productos en cartas:

```
Card(
  modifier = Modifier
    .fillMaxWidth()
    .padding(10.dp)
    .clickable { },
  shape = RoundedCornerShape(8.dp),
  elevation = CardDefaults.cardElevation(defaultElevation = 10.dp)
) { this: ColumnScope
  Column(
    horizontalAlignment = Alignment.CenterHorizontally,
    verticalArrangement = Arrangement.Center
  ) { this: ColumnScope
    Column(
      modifier = Modifier
        .padding(16.dp)
    ) { this: ColumnScope
      Text(
        it.name,
        style = MaterialTheme.typography.bodyLarge,
        fontWeight = FontWeight.ExtraBold,
        color = Color.White,
      )
      Spacer(modifier = Modifier.height(10.dp))
      Text(it.description)
      Spacer(modifier = Modifier.height(10.dp))
    }
  }
}
```

Figure 32: Card para productos

- Utilizamos una **Sealed Class** para representar una jerarquía de clases que heredan de una clase padre:

```
sealed class Items_menu_Admin(
    val icon: Int,
    val title: String,
    val ruta: String
) {
```

Figure 33: Sealed Class para los productos

- Para la barra de navegación ocupamos **objects**, para poner pasar de una ventana a otra de manera eficaz:

```
object Pantalla1Admin: Items_menu_Admin(
    R.drawable.usuariosadmin,
    title: "Usuarios", NavScreenAdmin.UsuariosAdmin.name)
@ LizetteTrejo
object Pantalla2Admin: Items_menu_Admin(
    R.drawable.cart,
    title: "Productos", NavScreenAdmin.ProductosAdmin.name)
@ LizetteTrejo
object Pantalla3Admin: Items_menu_Admin(
    R.drawable.categoriasadmin,
    title: "Categorias", NavScreenAdmin.CategoriasAdmin.name)
@ LizetteTrejo
object Pantalla4Admin: Items_menu_Admin(
    R.drawable.pedidos,
    title: "Pedidos", NavScreenAdmin.PedidosAdmin.name)
```

Figure 34: Objects para la barra de navegación

- Con ayuda de una **Enum Class** creamos un conjunto de valores que usan como identificador un nombre:

```
enum class NavScreenAdmin {
    VentanaInicio,
    UsuariosAdmin,
    ProductosAdmin,
    CategoriasAdmin,
    PedidosAdmin
}
```

Figure 35: Enum class para la barra de navegación

- De igual forma ocupamos una **Card** para las ubicaciones:

```
label = {
    Text(text = "E-mail")
},
leadingIcon = {
    IconButton(onClick = { /*TODO*/ }) {
        Icon(
            imageVector = Icons.Filled.Email,
            contentDescription = "Email icon"
        )
    }
},
```

Figure 36: Card para ubicaciones

- Con un **FAB** (Floating Action Button) agregamos las nuevas ubicaciones:

```
FloatingActionButton(
    onClick = {
        alias.value = ""
        address.value = ""
        reference.value = ""
        isEditing.value = false
        showDialog.value = true
    },
    modifier = Modifier
        .padding(16.dp)
        .align(Alignment.BottomEnd)
) {
    Icon(
        painter = painterResource(id = R.drawable.plus),
        contentDescription = "Agregar ubicacion",
        tint = Color.White
    )
}
```

Figure 37: FAB para ubicaciones

- Ocupamos **LifecycleOwner** para lanzar una nueva corrutina vinculada al ciclo de vida del componente:

```
lifecycleOwner.lifecycleScope.launch { this: CoroutineScope
    if(isEditing.value){
        handleBtnEdit(deliveryActive.value, alias.value, address.value, reference.value, mContext, loadingState, showDialog, dataDeliveryLocation)
        alias.value = ""
        address.value = ""
        reference.value = ""
        isEditing.value = false
    } else {
        handleBtnAdd(alias.value, address.value, reference.value, mContext, loadingState, showDialog, dataDeliveryLocation)
        alias.value = ""
        address.value = ""
        reference.value = ""
    }
}
```

Figure 38: LifecycleOwner para ubicaciones

- Lanzamos una corrutina **Launch** que llama a la función **getUsers** para obtener información de usuarios:

```
LaunchedEffect(Unit){ this: CoroutineScope
    lifecycleOwner.lifecycleScope.launch { this: CoroutineScope
        getUsers(loadingState, mContext, userState)
    }
}
```

Figure 39: Corrutina para los Usuarios del Admin

- Haciendo uso de **Retrofit** podemos realizar llamadas a una API para cada sección:

```
@GET("/api/users")
suspend fun getUsers(): UserResponse

@POST("/api/login")
@FormUrlEncoded
suspend fun login(
    @Field("email") email: String,
    @Field("password") password: String
) : AuthResponse

@POST("/api/signup")
@FormUrlEncoded
suspend fun signUp(
    @Field("name") name: String,
    @Field("email") email: String,
    @Field("password") password: String,
    @Field("last_name") last_name: String? = null,
) : SignUpResponse
```

Figure 40: Llamadas a API

- Creamos un **Object** para almacenar la URL base de una API:

```
object Config {  
  
    const val API_URL = "https://moviles.rmo.industries"  
  
}
```

Figure 41: Object de la URL de la API

## 5 Conclusión

La construcción de una aplicación móvil para pedidos de Food-Truck se presenta como una solución innovadora y relevante para abordar las necesidades cambiantes de los usuarios y la dinámica de los negocios de Food-Truck. Los puntos principales que hemos explorado destacan la importancia de brindar una experiencia de pedido conveniente y satisfactoria para los clientes, así como de optimizar la eficiencia operativa para los propietarios de Food-Truck.

Este proyecto aborda aspectos esenciales de seguridad, eficiencia y satisfacción del cliente. La introducción de tecnología móvil en el negocio de Food-Truck tiene el potencial de mejorar significativamente la experiencia del usuario al permitir realizar pedidos que puedan satisfacer sus necesidades de compra así como el uso de una aplicación móvil.

La ubicación estratégica de los Food-Trucks en zonas pobladas, como escuelas, campus y edificios gubernamentales, subraya la necesidad de ofrecer opciones de comida en lugares de alta afluencia de personas. La consideración de pedidos y entregas a pie y en bicicleta dentro de una zona predefinida refleja la necesidad de una logística de entrega eficiente y sostenible. La inclusión de opciones de pago con tarjeta, previo a la entrega, y la posibilidad de pagar al recibir el paquete muestra la importancia de adaptarse a las diversas preferencias de pago de los clientes.

Sin embargo, para lograr el éxito en este empeño, es crucial abordar aspectos clave, como la seguridad de datos y pagos, la comunicación efectiva entre usuarios y Food-Trucks, la optimización de rutas de entrega y la flexibilidad de horarios. Además, la retroalimentación de los usuarios será fundamental para la mejora continua del servicio.

En última instancia, este proyecto busca fusionar la comida de Food-Trucks con la conveniencia tecnológica para brindar una experiencia de pedidos eficaz.

## 6 Referencias Bibliográficas

### References

- [CareerFoundry. 2019] *What Is UX Research, And What's Its Purpose?*.  
<https://www.youtube.com/watch?v=6ZvEIdDGy2Q&list=PL4GEkVtNYG1IcHp91PMD9KHM1fc-Jbyjp&index=6>
- [CareerFoundry. 2019] *The Top 10 UX Research Tools You Need For User Research*.  
<https://www.youtube.com/watch?v=ejeRNNA1YB0&list=PL4GEkVtNYG1IcHp91PMD9KHM1fc-Jbyjp&index=7>
- [Grass, Jonny. 2021] *What is an Affinity Map? (And How to Make One)*.  
<https://careerfoundry.com/en/blog/ux-design/affinity-map/>
- [Veal, Raven. 2023] *How to Define a User Persona*.  
<https://careerfoundry.com/en/blog/ux-design/how-to-define-a-user-persona/>