
# Universidad Nacional Autónoma de México
## Facultad de Ingeniería

Cómputo Móvil
**Asignatura**
Semestre: **2022-2**

**Fecha: 28 de mayo, 2022**
**Grupo: 02**

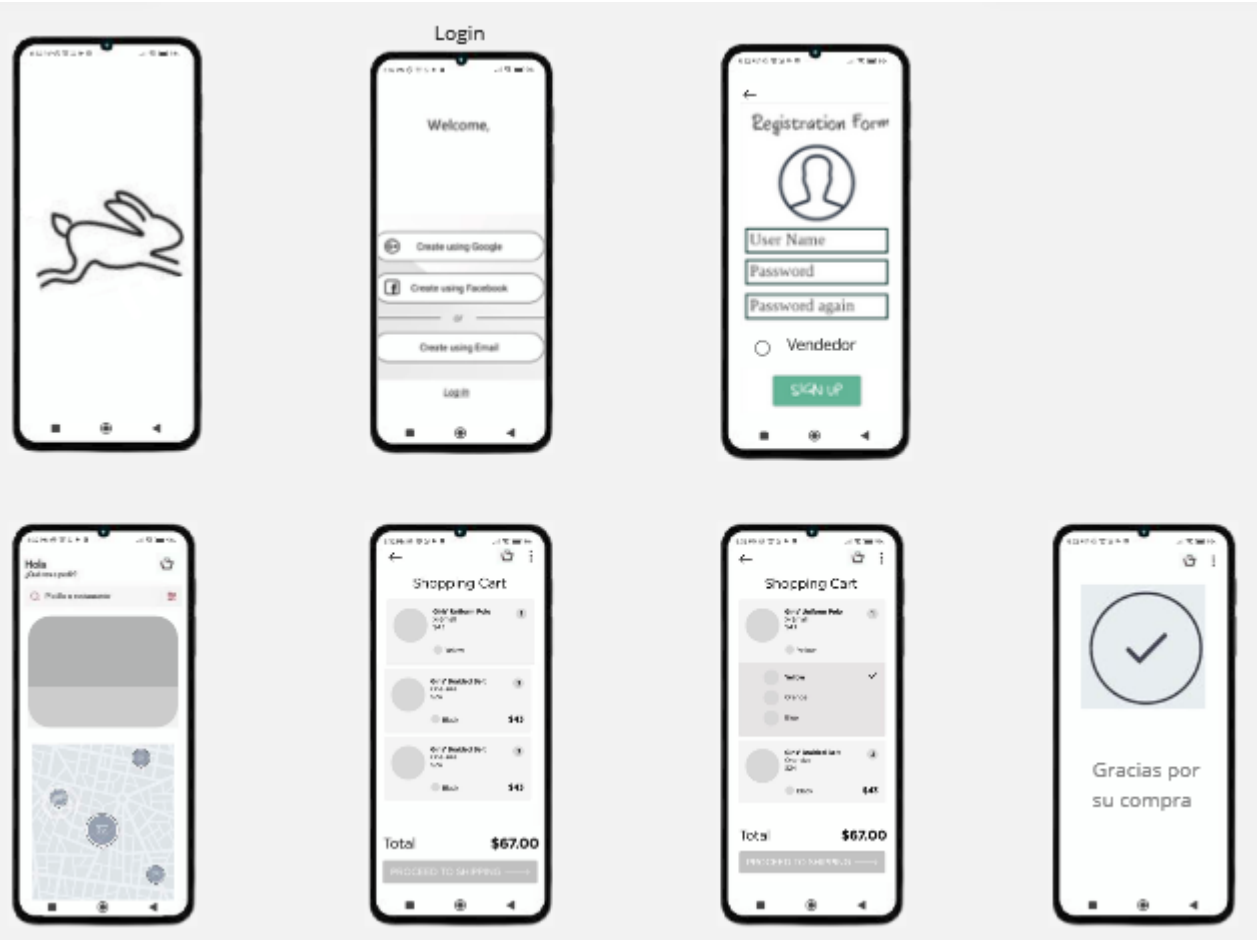
# Tercera evaluación parcial

**Erick Rodrigo Minero Pineda**
**Alumno**
**Hector Mauricio Garcia Serrano**
**Alumno**
**Laura Fabiola Aguilar Plascencia**
**Alumno**
**Yanni Martínez Martínez**
**Alumno**

**Ing. Marduk Pérez de Lara Domínguez**
**Profesor**

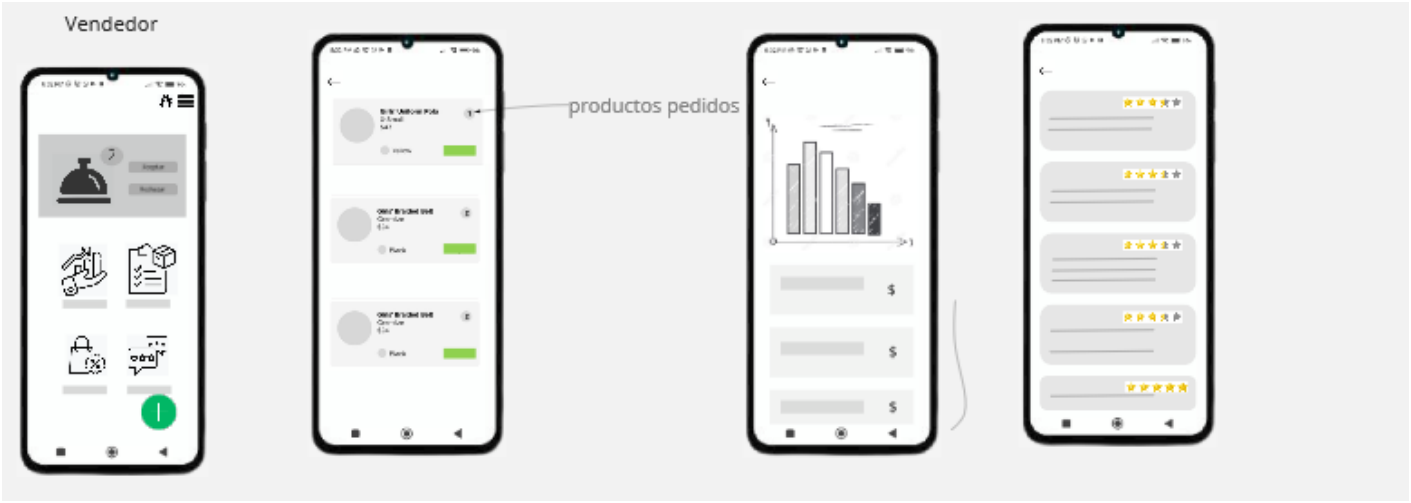
# Índice

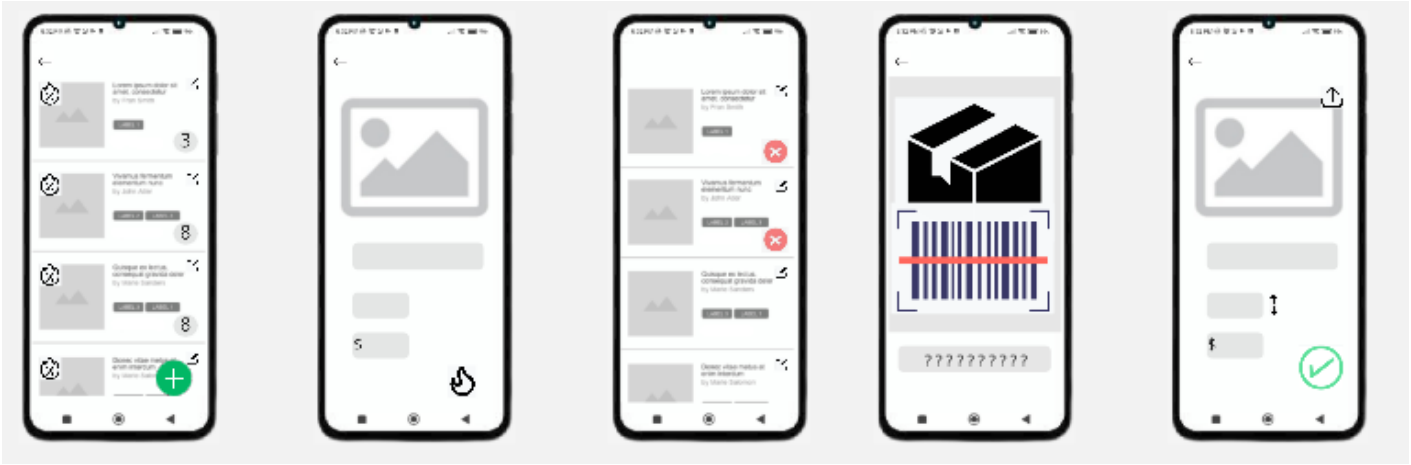
# Wireframes


Login
Welcome,
Registration Form
User Name
Password
Password again
Vendedor
SIGN UP
Shopping Cart
Total
$67.00
Shopping Cart
Total
$67.00
Gracias por
su compra

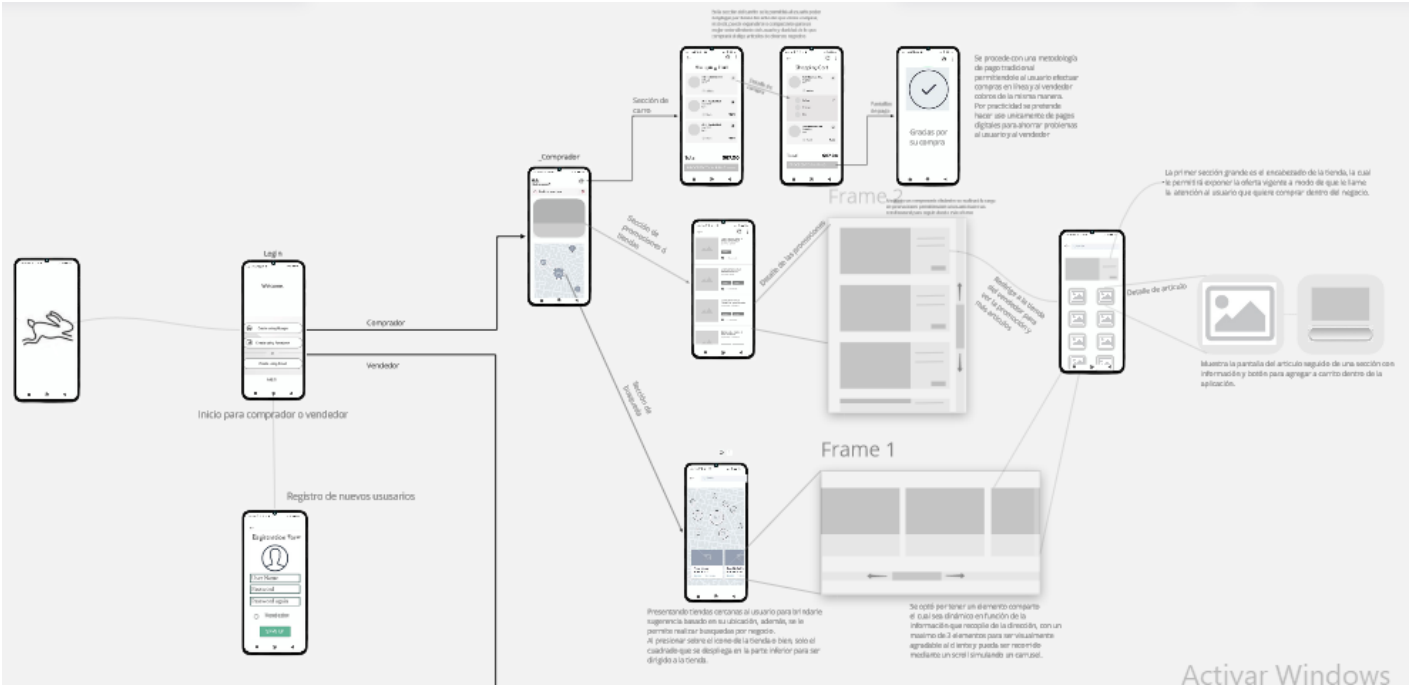

Frame 5


Frame 4

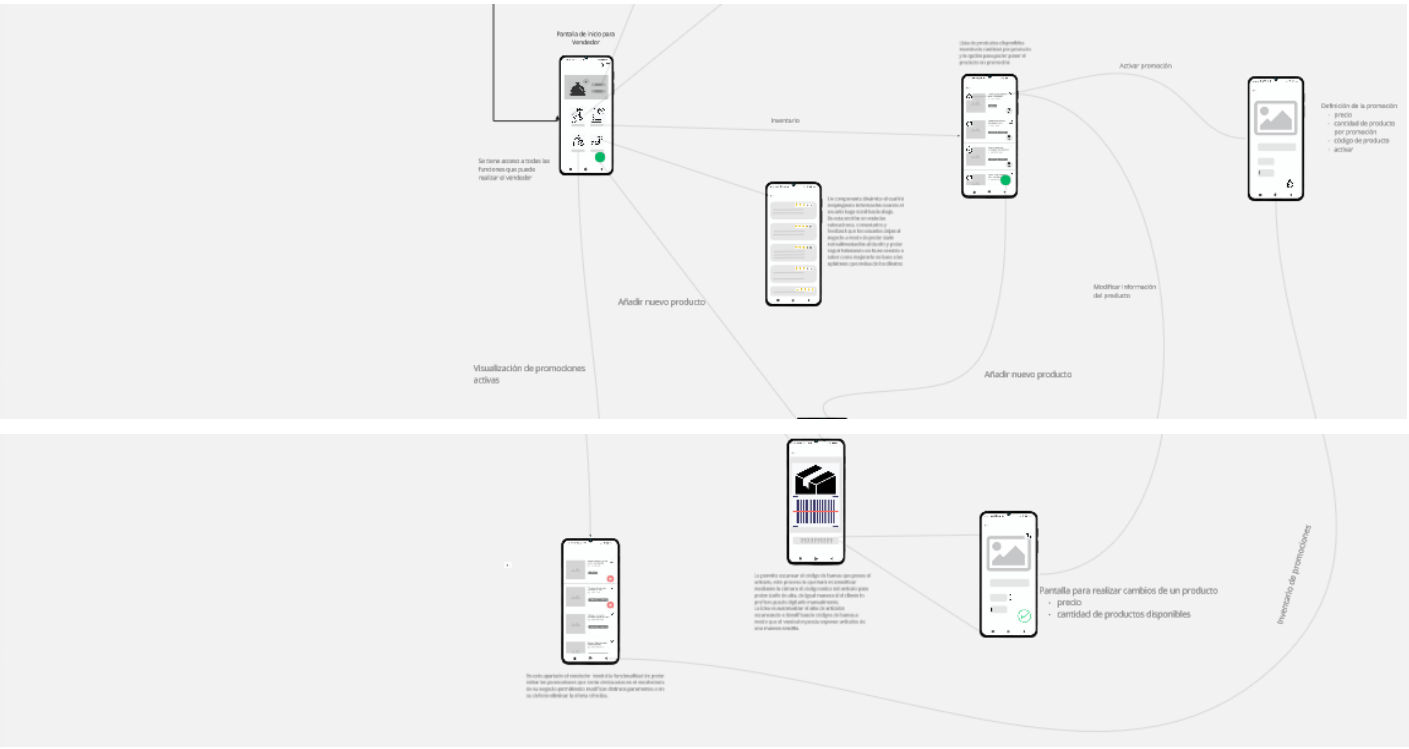

Vendedor
productos pedidos


??????????

# Flujo


Login
Comprador
Vendedor
Inicio para comprador o vendedor
Registro de nuevos ususarios
Sección de carro
Sección de promociones o tiendas
Sección de búsqueda
Frame 1
Frame 1
Gracias por su compra
Se procede con una metodología de pago tradicional permitiendole al usuario efectuar compras en línea y al vendedor cobros de la misma manera. Por practicidad se pretende hacer uso unicamente de pagos digitales para ahorrar procesos al usuario y al vendedor
La primer sección grande es el encabezado de la tienda, la cual le permitirá exponer la oferta vigente a modo de que le llame la atención al usuario que quiere comprar dentro del negocio.
Detalle de artículo
Muestra la pantalla del artículo seguido de una sección con información y botón para agregar a carrito dentro de la aplicación.
Presentando tiendas cercanas al usuario para brindarle sugerencia basado en su ubicación, además, se le permite realizar busquedas por negocio.
Al presionar sobre el icono de la tienda o bien, sobre el cuadrado que se despliega en la parte inferior para ser dirigido a la tienda.
Se optó por tener un elemento compartido cual sea dinámico en función de la información que recopile de la dirección, con un maximo de 3 elementos para ser visualmente agradable al cliente y pueda ser recorrido mediante un scroll simulando un carrusel.
Activar Windows


productos predilectos
Desglose de predilectos identificados por nombre la persona que realiza el pedido y los productos que tiene que preparar para la entrega
Modificaciones de nuevos pedidos
Ventas generadas



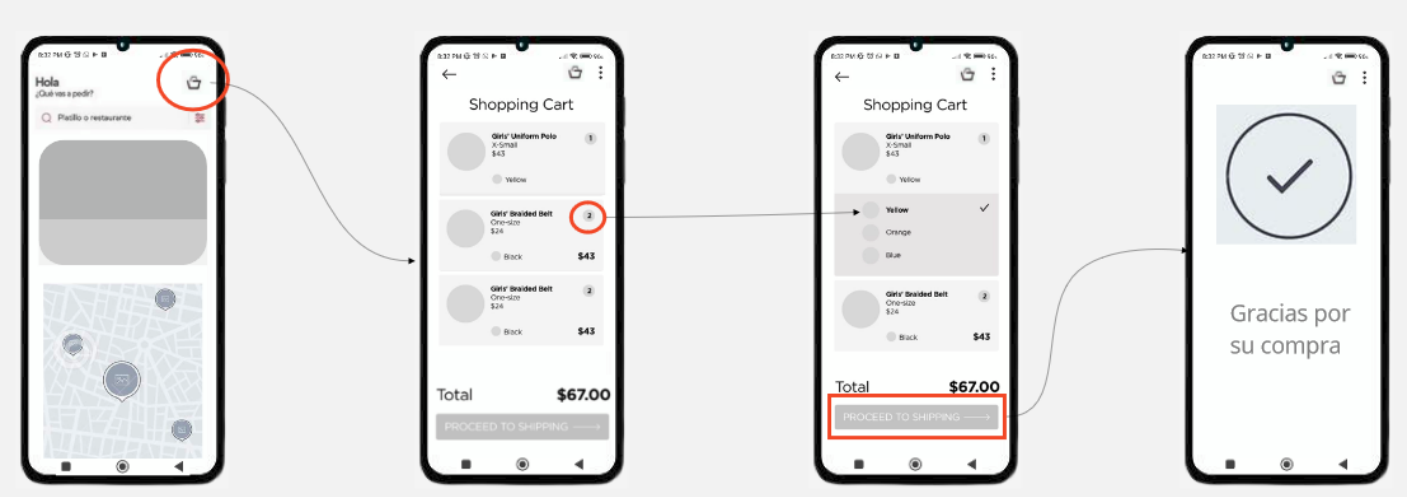

## Gestos

- Se emplearán gestos tradicionales de scroll vertical y horizontal para visualizar el contenido de los contenedores dinámicos.
- La navegación tradicional será empleada permitiendo al usuario regresar de pantalla mediante un botón ubicado en la parte superior izquierda.

## Funcionalidades de cada pantalla

### Carrito de compras

- Despliegue de productos correspondiente a cada negocio
- Pago total de productos

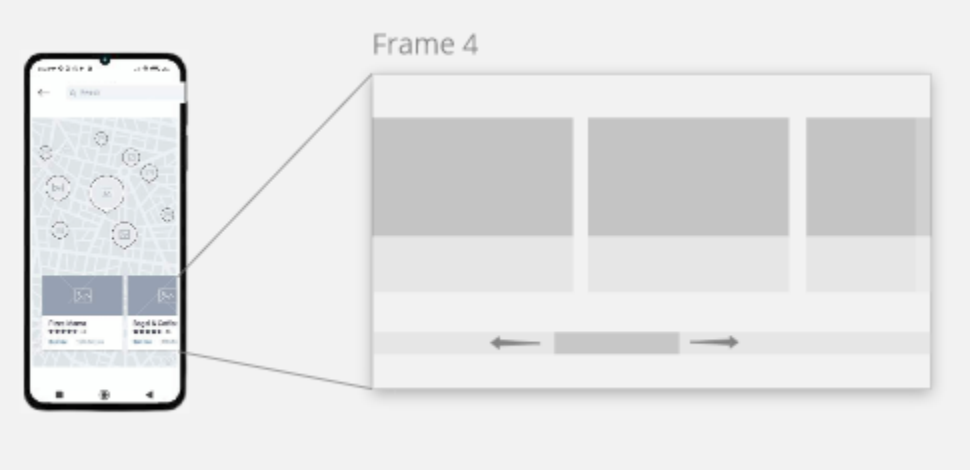


## Visualización de promociones

- Muestra las promociones activas de cada negocio
- Redirige al negocio al que corresponda dicha promoción al dar clic
- Al posicionarse en un producto muestra el precio y la opción de añadir al carrito

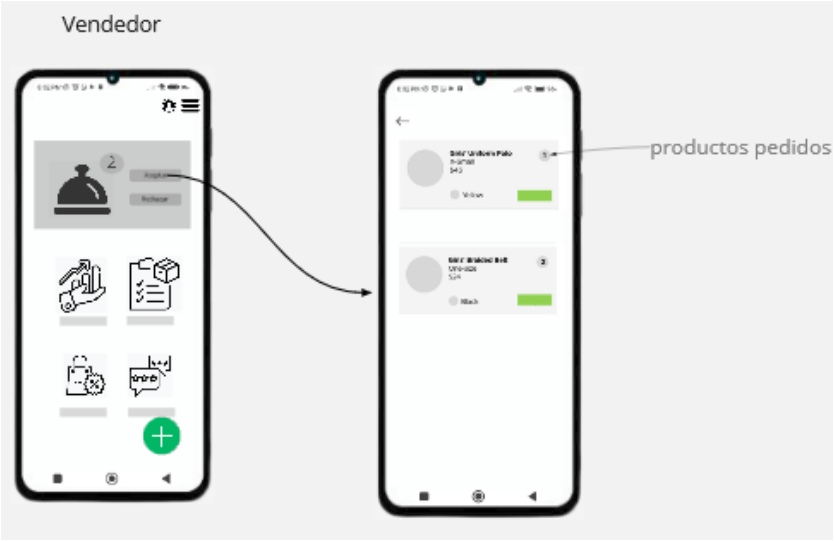
## Ubicación de negocios

- Se muestran los locales que se han registrado en la app dentro del mapa
- Se muestran los negocios con su tipo de producto/servicio.




## Notificaciones nuevos pedidos

- Muestra la cantidad de pedidos pendientes por preparar/realizar.
- Opción de dar ‘hecho’ para indicar al usuario que su compra ya está preparada.

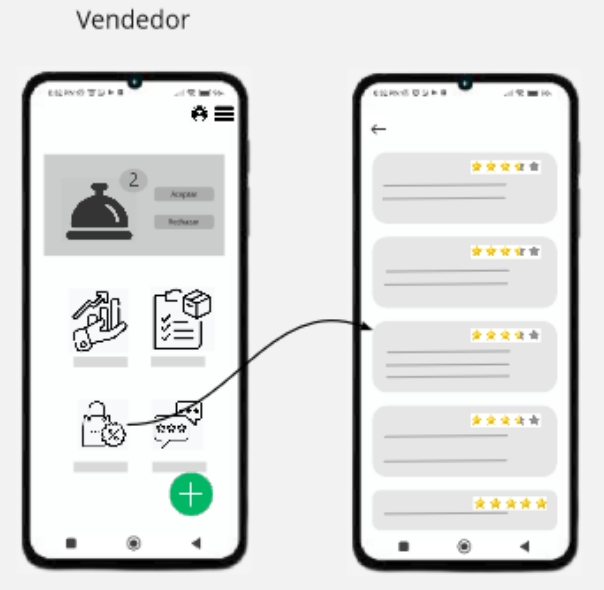



## Ventas generadas

- Se muestra un gráfico estimado sobre todas las ventas que ha tenido el vendedor durante la semana.
- Se muestra un histórico de ventas realizadas durante el mismo lapso de tiempo de una semana permitiéndole llevar un control de las ventas que ha realizado y monto total.

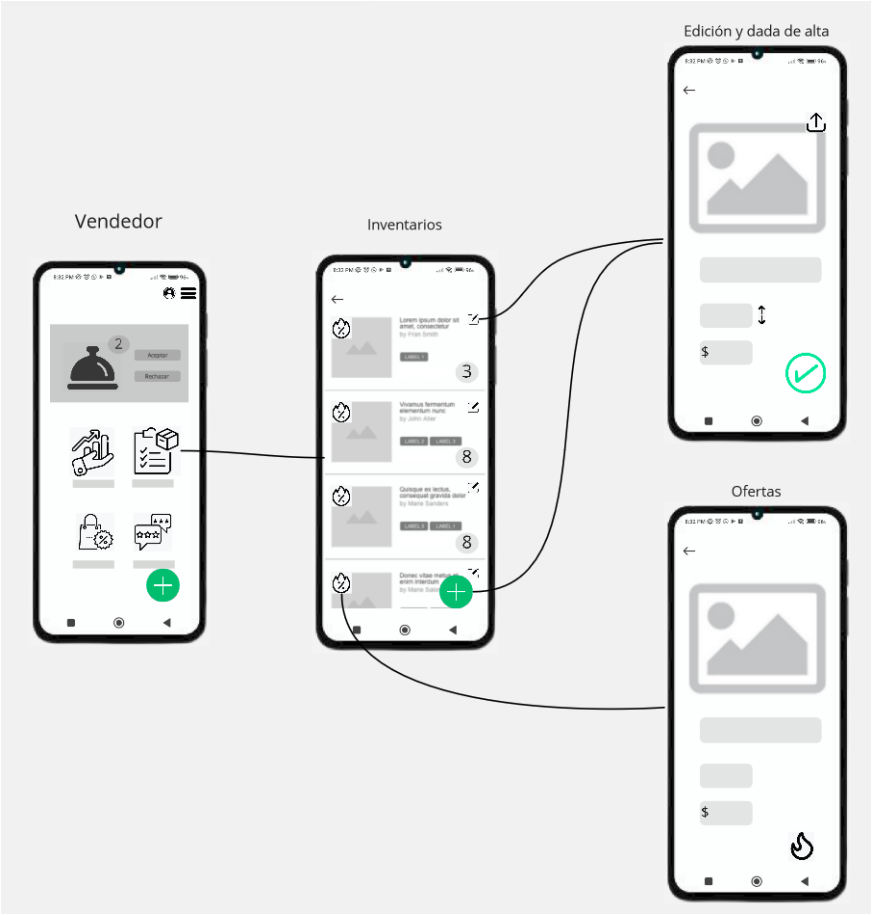

## Opiniones

- Visualización de comentarios
- Calificaciones de usuarios respecto al producto/servicio



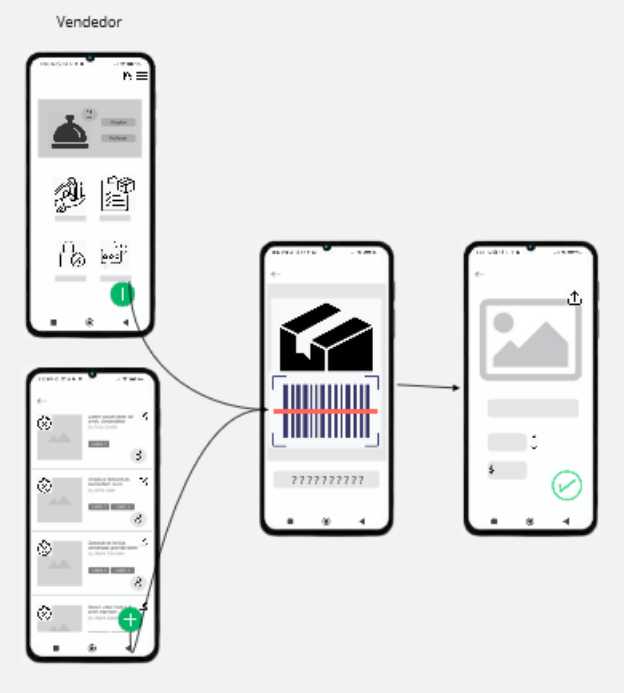

## Control de inventarios

- Al presionar el botón de inventario el vendedor es capaz de ver el inventario que tiene, algunas de las funcionalidades que tiene a su disposición son:
    - Editar articulo: Descripción, imagen, precio y si es aplicable o no a una oferta.
- Capacidad de destacar un artículo como oferta para que destaque en el encabezado del negocio y sea más llamativo para el cliente.
- Opción de dar de alta un artículo, indicando nombre, precio, imagen y cantidad disponible



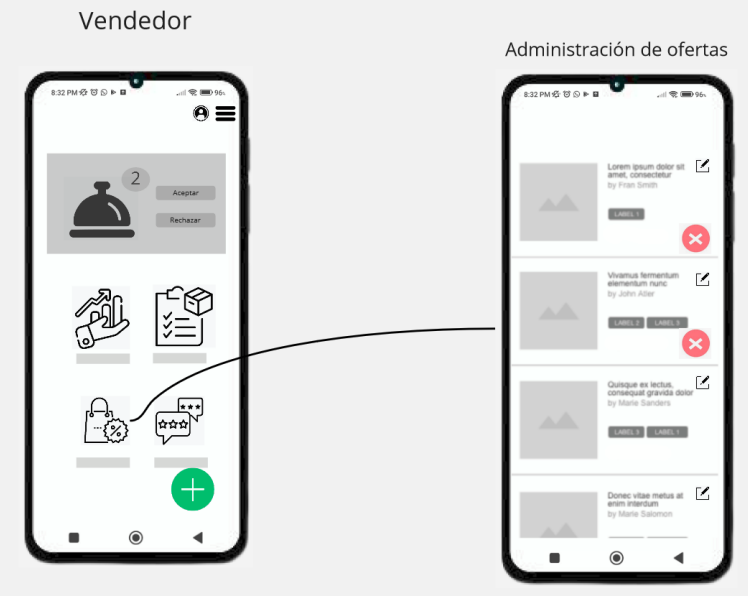
## Agregar producto nuevo

- Escaneo de código de barras mediante cámara.
- Agregar información relevante del producto (precio, cantidad disponible, etc)



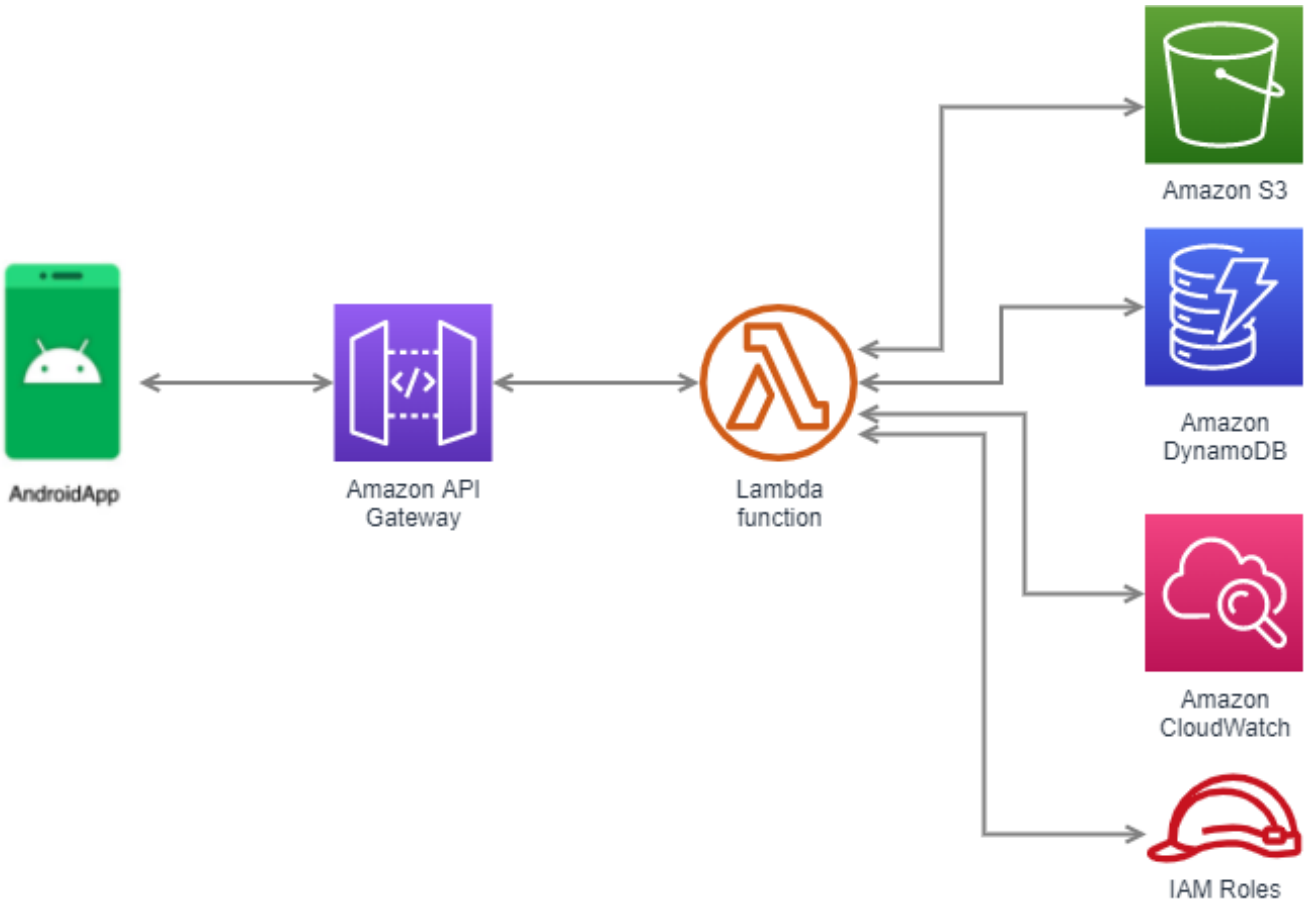

## Promociones activas (vendedor)

- Muestra las promociones o descuentos que haya indicado el negocio
- Se puede modificar la promoción o eliminarla



# Back-end

Principalmente la aplicación se estará conectando a una API REST. Creemos que nuestra aplicación puede existir sin servidor propio utilizando 100% servicios cloud, es por ello que se optó por contratar los servicios de Amazon Web Services, pensando también en la facilidad que estos otorgan para escalar el producto de manera rápida y sencilla, además, mientras no seamos tan grandes únicamente estaremos pagando por lo que se consume, sin tener que hacer inversiones fuertes en infraestructura en un inicio.
En primera instancia se desarrolló el siguiente diagrama de cómo sería esta conexión y qué servicios se requieren para tener nuestro backend sin servidor:




El diagrama es una arquitectura sencilla de AWS donde estaremos utilizando lo servicios de:

- Amazon API Gateway para que facilite la publicación, mantenimiento, supervisión y funcionamiento de la API de manera escalable y administre las conexiones. Se levantará el servicio de API HTTP, mismo que es económico.
- El servicio de Lambda: este es un servicio que permite ejecutar código sin servidor (funciones) y por petición, tiene soporte para Node.JS, Python, Go, .NET y Java.
  En este servicio se va a subir y ejecutar el código de cada operación Create, Read, Update, Delete (CRUD) que requiera nuestra app, se creará una función lambda por operación y esto por tabla (4 funciones por tabla) ya que es la manera más recomendable de hacerlo, pero claramente se puede crear una sola función por tabla que contenga las 4 operaciones.
- Amazon DynamoDB: es un servicio de base de datos NoSQL de clave-valor sin servidor que ofrece Amazon. La ventaja de utilizar este servicio es que ya trae soporte para integrar con otros servicios de AWS como los buckets de S3, CloudWatch, CloudTrail, entre otros.

Utilizaremos DynamoDB para crear nuestras tablas de datos y las operaciones CRUD que se hagan sobre las tablas serán administradas por las Funciones Lambda.

- Amazon S3: es un servicio de almacenamiento en la nube, similar a como funciona google drive pero de manera profesional. En este servicio guardaremos recursos como son imágenes, videos, datos de la aplicación, respaldos y en un futuro análisis de nuestros datos. Al ser un servicio de AWS es fácil de integrar con nuestras tablas en DynamoDB, así si algún registro clave-valor tiene como valor una imágen o video, podemos linkearlo a su ubicación en S3.
- Amazon CloudWatch: este servicio es un monitor de infraestructura en la nube de AWS (recursos y aplicaciones). Lo utilizaremos para monitorear las funciones Lambda, esto nos permitirá ver errores que se puedan producir en ellas. Además podremos monitorear nuestras tablas hechas en DynamoDB.
- Amazon IAM: es el servicio encargado de los accesos a las aplicaciones y recursos que levantamos en AWS. Con él podemos crear usuarios y roles que nos permiten una mayor seguridad e integridad de nuestros recursos y servicios creados en la nube, así como para administrarlos correctamente, en pocas palabras, sirve para definir qué usuarios pueden meter mano a qué recursos.

En resumen:

- Nuestra aplicación tendrá definidos “end points” creados mediante API Gateway y Lambda. Estos serán administrados dentro de Android con las bibliotecas de retrofit y gson converter todo siguiendo una arquitectura Model View ViewModel (MVVM) que permita una fácil comunicación entre las vistas y la interacción con el almacenamiento tanto remoto como local
- Con API Gateway, además de crear los endpoints, se utiliza para poder hacer peticiones con el protocolo de transferencia HTTP (GET, PUT, POST, DELETE), estas operaciones serán programadas por separado en Python utilizando las librerías Fast API, Flask o Django Rest Framework (esto queda a consideración del equipo de desarrollo web) y ejecutadas por el servicio de Lambda (sin servidor)
- A su vez, las operaciones ejecutadas en este servicio estarán afectando a las tablas de la base de datos hechas en DynamoDB.
- La comunicación entre la aplicación y la API REST es en formato JSON ya que este formato es reconocido por todos los servicios mencionados; desde la aplicación con retrofit, pasando por la API Gateway, el código Python en Lambda hasta los documentos almacenados en la base de datos, pues al ser Nosql su formato es JSON (clave-valor).

Los modelos de datos involucrados son los siguientes:

- Usuario (vendedor y comprador):
    - Identificador
    - Nombre
    - Correo electrónico
    - Edad
    - Dirección
    - Foto de perfil
    - Tipo (vendedor/comprador/administrador)
    - Datos de pago
- Producto/artículo:
    - Identificador

    - Nombre
    - Categoria
    - Descripción
    - Precio
    - Imagen
    - Codigo
    - Tienda asociada
- Promoción:
    - Identificador
    - Código de promoción
    - Vigencia
    - Descuento
    - Artículo asociado
    - Tienda asociada
- Tienda/negocio:
    - Identificador
    - Nombre
    - Imagen/Logotipo
    - Dirección
    - Usuario/vendedor
    - RFC
    - Horario
    - Teléfono(s) de contacto
    - Correo electrónico
- Feedback:
    - Usuario
    - Score
    - Comentario
    - Tienda asociada
- Pedido/Compra:
    - Identificador
    - Cantidad de artículos
    - Artículos
    - Monto total
    - Fecha-Hora de compra
    - Tienda asociada/vendedor
    - Identificador comprador
    - Identificador tienda
    - Datos de facturación

Para cada uno se deberá crear su respectiva tabla en DynamoDB y su conjunto de funciones Lambda para las 4 operaciones fundamentales (CRUD), claro que puede ser más de una, en el caso de la operación Read que se traduce a un GET, se puede hacer de todos los ítems o filtrando por ID, esa es una de las ventajas que tienen las API, podemos filtrar desde BackEnd y ahorrarse ese proceso en el código de la aplicación, solo mostrarlo.

A continuación se muestra el detalle de los costos estimados del servicio de backend de tener la aplicación en etapa operativa.

| Detalle S3 | USD | | |
|---|---|---|---|
| Servicio | Costo | Cantidad | Total |
| Almacenamiento | 0.023 | 20 | 0.46 |
| PUT requests | 0.000005 | 5000 | 0.025 |
| GET requests | 0.0000004 | 10000 | 0.004 |
| Data returned GB | 0.0007 | 5 | 0.0035 |
| Data scanned GB | 0.002 | 5 | 0.01 |
| Total | | | 0.5025 |

| Detalle API Gateway | USD | | |
|---|---|---|---|
| Servicio | Costo | Cantidad | Total |
| HTTP API requests units | 0.000001 | 5000000 | 5 |
| 5,120 KB per request / 512 KB request increment = 10 request(s) | | | 10 |
| | | | |
| Total | | | 50 |

| Modalidad | Servicio | Costo Unitario | Unidades | Total |
|---|---|---|---|---|
| Mensual | Nube | | | |
| Consulta | AWS API Gateway | 50 | 1 | 50 |
| Almacenamiento | AWS S3 | 0.5025 | 10000 | 5025 |
| Costo operativo de aplicación | | | | 5075 |

En conclusión, los gastos estimados de la operación mensual de la aplicación es de $99,374.44 MXN

## Almacenamiento local

De almacenamiento local casi no se tendrá, puesto que la aplicación requiere acceso a internet para tener sentido. Por el lado del usuario vendedor, este puede tener más interacción con la aplicación que no requiera estar conectándose a los servicios en la nube constantemente, se tiene considerado que la petición se haga una vez y parte de la información como el inventario, promociones activas y sobre todo las estadísticas de venta se almacenan localmente utilizando las distintas maneras que Android proporciona para esto, se tiene pensado en un inicio que se utilice Jetpack Data Store quedando a consideración del equipo de desarrollo. Para el usuario común se almacenará en el almacenamiento local algunas imágenes de productos y promociones.
Para todos los usuarios también se almacenarán datos de autenticación y datos de personalización (por ejemplo, si llegamos a tener un modo oscuro esta preferencia sea almacenada), al almacenar en local las configuraciones de usuario podemos garantizar una mejor fluidez a la hora de ingresar a la app, ya que no requeriría realizar muchas peticiones al inicio.

# Dispositivos soportados

El tamaño de la pantalla es el espacio visible proporcionado por la IU de la app. El tamaño de la pantalla no es el tamaño real de la pantalla del dispositivo; se debe considerar la orientación de la pantalla y decoración del sistema Al ser una aplicación dirigida a un sector de no tan amplios recursos; negocios micro, pequeños y medianos se tiene pensado que la aplicación esté disponible para un gran porcentaje de dispositivos móviles, así que, como ya se ha mencionado, la versión mínima soportada es Android 5.0 hasta la última que es Android 12.0.

En general, planeamos que además de tener soporte a pantallas de smartphones en general, se tenga para tabletas con sistema operativo Android, para que los negocios que encuentren oportunidad de crecimiento en nuestra app puedan utilizar el dispositivo que más les convenga en su negocio.

Es bien sabido que con el evolucionar de los sistemas operativos Android el tamaño de las pantallas va en aumento. En Android se tienen, en general, cuatro tamaños existentes: pequeño, normal, grande y muy grande. Y varias densidades: mdpi (media), hdpi (alta), xhdpi (muy alta), xxhdpi (muy muy alta) y otras. De forma predeterminada, la app a desarrollar ya es compatible con todos los tamaños y densidades de pantalla porque el sistema realiza los ajustes adecuados al diseño y recursos de imagen según sea necesario para cada pantalla. Sin embargo, vamos a optimizar la experiencia de usuario en cada pantalla agregando diseños especializados para diferentes tamaños de pantalla e imágenes de mapa de bits optimizadas para densidades de pantalla comunes.

# Sensores

La aplicación trabaja con base en la ubicación del usuario, misma que puede ser agregada manualmente o aprovechar el GPS, con la idea de mostrar lo más cercano al usuario. Para ello se hará uso de la API de Google Maps, pero vamos a customizar el mapa poniendo nuestra capa de personalización sobre la vista del mapa para una mejor experiencia de usuario, como se hace en otras aplicaciones como Rappi o Uber Eats.


Al momento de registrar productos se hace uso de la cámara como si fuera un lector de código de barras y de QR, simplemente se abre la cámara, se toma la imágen y mediante

código se procesa la imagen obteniendo de ella un código numérico para posterior registro del producto. Para la parte de la cámara no utilizaremos la aplicación por defecto del sistema, haremos nuestro propio diseño para una mejor experiencia de usuario.

## Equipo y roles de trabajo

| Rol |
| --- |
| Project Manager |
| Business Analist |
| Arquitecto de soluciones |
| Android Developer Sr |
| Android Developer Jr |
| Desarrollador Web Full Stack Sr |
| Desarrollador Web Full Stack Jr |
| Tester |
| DBA |
| Diseñador UI/UX |

El equipo está un poco alineado a las metodologías de desarrollo Scrum y Ágil, esto para aplicar alguna de ellas al momento del desarrollo y tener un orden en la realización del proyecto.
Para comenzar con el proyecto requerimos diseñadores UI/UX que nos ayuden a mejorar los Wireframes presentados en esta propuesta y llevar a cabo prototipos que nos guíen, esto con el objetivo de que en algún momento aprobemos un diseño final y comience el desarrollo en Android y BackEnd.
La persona con el rol de Arquitecto de soluciones será el encargado de definir claramente la arquitectura de servicios AWS, administrador de los mismos y creador de los roles para acceder a configurar los servicios, su comunicación principal es con el project manager, nosotros y los desarrolladores web.
Entre los miembros del equipo se planea incluir personas con poca experiencia (Jr) ya que nos ayuda a reducir costos, pero procurando nuestro producto, se considera tener al menos una persona experta (Sr) en cada rol para que acompañen a los principiantes y con ello no descuidar la productividad del equipo y la calidad final del producto.
Consideramos incluir al menos una persona para el rol de Tester para que cada sprint o versión que se termine sea aprobada por este rol, es una parte importante para evitar muchos remakes o detenimientos a la hora del desarrollo.
Las personas de web nos ayudarán principalmente con el desarrollo de todos los web services que se requieran como la API REST, este equipo estará constantemente en comunicación con el arquitecto de soluciones y el equipo de desarrollo Android para que la comunicación entre la aplicación y los servicios sea la correcta.

# Tiempo y costos

El desarrollo de la aplicación será de 8 meses, durante los cuales se tienen considerados los siguientes gastos.

## Gastos indirectos

| Gastos | | | |
|---|---|---|---|
| Modalidad | Concepto | Cantidad (MXN) | **Unidades** |
| Mensual | Internet | $700.00 | 8 |
| Mensual | Electricidad | $200.00 | 8 |
| Licencia | Zoom | $150.00 | 1 |
| Mensual | GitHub Team | $2,303.51 | 8 |
| | | | |
| | Gastos (MXN) | | $25,778.08 |

## Personal

| Rol | Salario mensual (MXN) | # de recursos | ISR/ISPT | Salario total mensual (MXN) |
|---|---|---|---|---|
| Project Manager | $40,000 | 1 | $7,384 | $47,384 |
| Business Analist | $30,000 | 1 | $5,032 | $35,032 |
| Arquitecto de soluciones | $35,000 | 1 | $6,208 | $41,208 |
| Android Developer Sr | $32,000 | 2 | $5,502 | $75,005 |
| Android Developer Jr | $14,000 | 1 | $1,549 | $15,549 |
| Desarrollador Web Full Stack Sr | $32,000 | 1 | $5,502 | $37,502 |
| Desarrollador Web Full Stack Jr | $13,000 | 1 | $1,349 | $14,349 |
| Tester | $13,000 | 2 | $1,349 | $28,698 |
| DBA | $24,000 | 1 | $3,685 | $27,685 |
| Diseñador UI/UX | $18,000 | 2 | $2,404 | $40,807 |
| | | | | |
| | Total acumulado mensual (MXN) | | $363,219 | |
| | Total acumulado proyecto (MXN) | | $2,905,748 | |

Dando un total de 2,931,526.08 MXN

## Gastos de desarrollo acelerado

Si analizamos los gastos de un desarrollo acelerado se estima que los gastos de personal aumentan un 20%

Por lo que tenemos:

| Rol | Salario mensual (MXN) | # de recursos | ISR/ISPT | Salario total mensual (MXN) |
|---|---|---|---|---|
| Project Manager | $48,000 | 1 | $7,384 | $55,384 |
| Business Analist | $36,000 | 1 | $5,032 | $41,032 |
| Arquitecto de soluciones | $42,000 | 1 | $6,208 | $48,208 |
| Android Developer Sr | $38,400 | 2 | $5,502 | $43,902 |
| Android Developer Jr | $16,800 | 1 | $1,549 | $18,349 |
| Desarrollador Web Full Stack Sr | $38,400 | 1 | $5,502 | $43,902 |
| Desarrollador Web Full Stack Jr | $15,600 | 1 | $1,349 | $16,949 |
| Tester | $15,600 | 2 | $1,349 | $16,949 |
| DBA | $28,800 | 1 | $3,685 | $32,485 |
| Diseñador UI/UX | $21,600 | 2 | $2,404 | $24,004 |
| | | | | |
| | Total acumulado mensual (MXN) | | $341,164 | |
| | Total acumulado proyecto (MXN) | | $2,729,311 | |

Dando un total de 3,515,880.48

## Costo operativo

Durante el desarrollo, ya sea de forma normal o acelerado, no se requiere ninguna capa de pago de los servicios de la nube pues con la capa gratuita tenemos los recursos suficientes para poder desarrollar plenamente. Para la etapa operativa los gastos ya se detallaron en la sección Back-End.

# Anexos

- Para mejor visualización de los wireframes y el flujo de pantallas consultar el siguiente link:

- App -Ay´ voy!
  https://miro.com/welcomeonboard/eWgyTDhZdHBlaFdIT0M1N3J0Nk9HTm5KVlFiMHNqeVY2OVBCSlE1TUhueERuR2lSZTVDUDF6Mkdkd0dHWGFNa3wzMDc0NDU3MzUwMzg5Mjg1NzAy?share_link_id=848512466116

# Referencias


- AWS Pricing Calculator. (s. f.). AWS. Recuperado 20 de mayo de 2022, de https://calculator.aws/#/
- Contapaqi. (2021). ISR o ISPT (Impuesto sobre el producto del trabajo): ¿cómo hacer su cálculo? CPI. Recuperado 21 de mayo de 2022, de https://www.contpaqi.com/publicaciones/contabilidad/como-realizar-el-calculo-del-ispt-impuesto-sobre-el-producto-del-trabajo
- El Contribuyente. (2022, 20 mayo). Calculadora de ISR. Recuperado 21 de mayo de 2022, de https://www.elcontribuyente.mx/calculadora/isr/
- Mi Negocio App. (s/f). Web.app. Recuperado el 23 de mayo de 2022, de https://goolaxo.web.app/