

BuyNote



Antonio Cordero Serrano 2º CFGS DAM
Creación de la presentación: 10/10/2025



Índice

- **Introducción (6)**
- **Plan de viabilidad (7 - 16)**
 - Necesidades actuales del mercado (8)
 - Competencia (9 - 12)
 - Valor diferencial de la app (13)
 - Plan de inversión (14)
 - Umbral de rentabilidad (15)
 - Plan de marketing (16)



Índice

- **Requisitos del sistema (17 - 31)**

- Requisitos de información (18 - 22)
- Requisitos funcionales (23 - 28)
- Casos de uso (29 - 31)

- **Modelo de datos (33 - 53)**

- Modelo conceptual (34 - 36)
 - Descripción de entidades (37 - 42)
 - Descripción de interrelacionales (43 - 44)
- Modelo lógico de datos (45 - 50)
- Consultas (51 - 54)

Índice

- **Interfaz de usuario (55 - 61)**

- Definición de pantallas (Figma) (56 - 59)
- Mapa de navegación (60 - 61)

- **Desarrollo del proyecto del sistema (62 - 66)**

- Fase 1: Funcionalidades a desarrollar y plan de pruebas (63 - 65)
- Fase N: Funcionalidades a desarrollar y plan de pruebas (66 - 67)



Índice

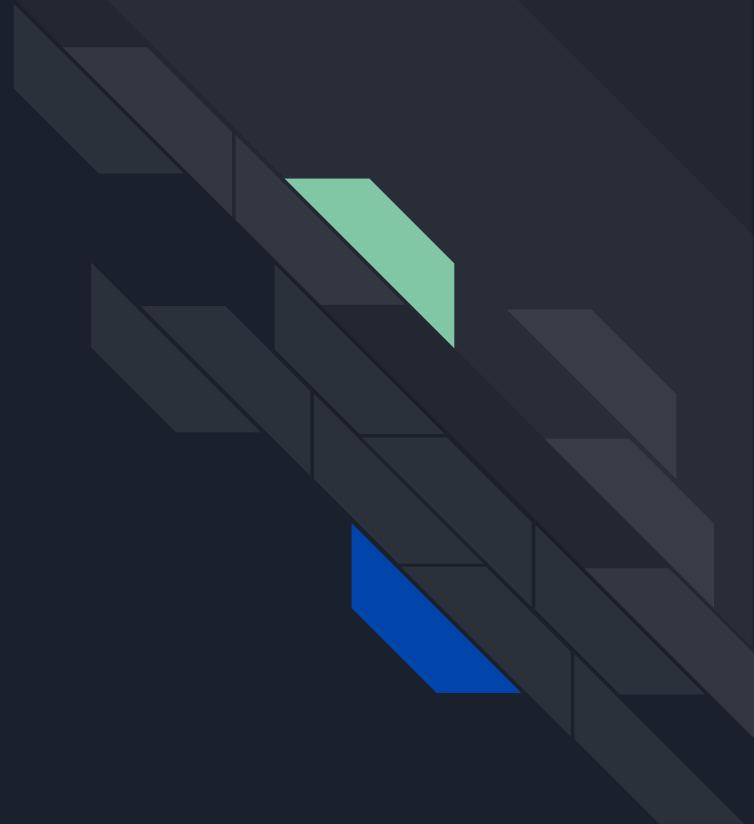
- **Información para el cliente (68 - 76)**
 - Requisitos físicos y de hardware (69)
 - Requisitos software (70)
 - Manual de instalación, configuración y administración (71 - 73)
 - Manual de usuario (74 - 76)
- **Bibliografía y fuentes de información (77)**



Introducción

Un software para crear y gestionar listas de la compra, tanto individuales como compartidas. Permite añadir productos, incluir observaciones, marcar compras completadas, registrar el precio total y dividir automáticamente el gasto entre los usuarios. En Android, los productos también pueden añadirse mediante código de barras para agilizar la captura de información.

Plan de viabilidad





Necesidades actuales del mercado

- Organización sencilla y rápida de las compras.
- Colaboración entre varios usuarios en tiempo real.
- Identificación clara de quién añade o compra cada producto.
- Control de gastos compartidos, algo poco explotado por la competencia.
- Sincronización multiplataforma (móvil y web).



Competencia

Keep

Ventajas

- Sincronización con cuenta Google
- Integración con Google Assistant
- Recordatorios geolocalizados

Desventajas

- No está pensada para listas de la compras
- Más orientada a notas rápidas



Competencia

Listonic

Ventajas

- Sugerencias automáticas al escribir
- Soporte en múltiples idiomas
- Añadir precios a los productos

Desventajas

- No realiza cálculo colaborativo
- No se refleja quien añadió/compró productos



Competencia

Out of milk

Ventajas

- Funciones como inventario de despensa
- Lista de tareas

Desventajas

- Interfaz “antigua”
- No tiene sincronización en tiempo real
- No tiene función de división de gastos



Competencia

Bring

Ventajas

- Diseño organizado y atractivo
- Categorías que hacen atractiva la organización

Desventajas

- Incluye publicidad constante
- No identifica quien añadió/compró los productos



Valor diferencial de la app

- Escaneo de productos por código de barras en Android.
- Cálculo automático de cuánto paga cada usuario, basado en el precio total de la lista.
- Registro de qué usuario añade y quién compra cada producto.
- Sincronización inmediata entre Android y Web.
- Coste de desarrollo y mantenimiento mínimo, lo que permite crecer sin grandes inversiones iniciales.



Plan de inversión

Costes fijos reales

- **Firestore (Firebase)** → gratuito (plan free).
- **Hosting (Vercel)** → gratuito.
- **Dominio web** → 12 €/año.
- **Marketing inicial** → redes sociales y ASO (estimación: 50–100 € iniciales)(opcional).

Inversión inicial real: aprox. 12-110€ el primer año, sumando el dominio y algo de publicidad básica.



Umbral de rentabilidad

- **Ingresos previstos:**
 - **Publicidad:** ingresos futuros mediante anuncios integrados.
 - **Suscripción premium:** 2,99 €/mes con funciones exclusivas (listas ilimitadas, eliminar anuncios, funciones premium).
- **Ejemplo de rentabilidad mínima:**
 - Con **5 usuarios premium** (2,99 €/mes), ya se generan **15 €/mes** → **180 €/año**, suficiente para cubrir el dominio y marketing básico.
 - El resto de ingresos, como publicidad, será beneficio adicional.

Esto significa que con menos de 10 usuarios premium al año se cubren todos los costes, lo que hace al proyecto viable incluso con una base pequeña de usuarios.



Plan de marketing

Estrategia

- **Fase inicial:** difusión en redes sociales (Instagram, TikTok) con vídeos mostrando cómo dividir gastos fácilmente o cómo organizar compras en grupo.
- **Versión gratuita atractiva:** ofrecer todas las funciones básicas para atraer usuarios y después destacar ventajas premium.

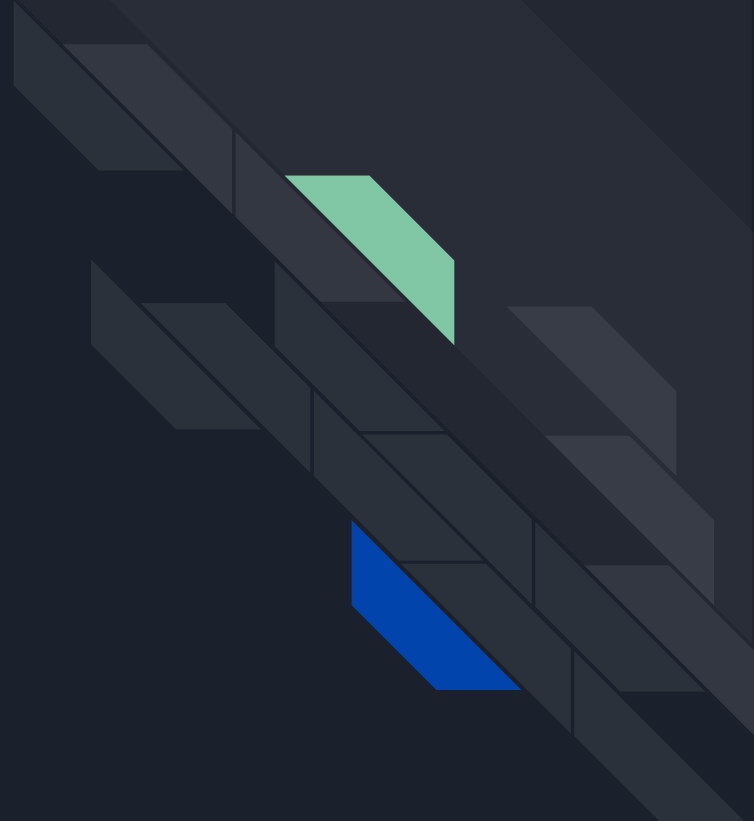
Público objetivo

- Familias que organizan la compra semanal.
- Compañeros de piso que comparten gastos.
- Amigos que organizan fiestas, barbacoas o viajes.

Plan a futuro

- Incluir publicidad ligera en la versión gratuita (modelo Freemium).
- Promociones de prueba del plan premium para atraer usuarios.
- Generar informes anónimos de tendencias de compra

Requisitos del sistema





Requisitos de información

Usuarios:

- Correo electrónico (para registro, login y compartición de listas).
- Contraseña (para autenticación).



Requisitos de información

Listas de la compra:

- Identificador único de la lista.
- Nombre de la lista.
- Estado (individual o compartida).
- Usuarios vinculados (propietario e invitados).



Requisitos de información

Productos:

- Nombre del producto.
- Marca y modelo (si se añade por código de barras).
- Estado (pendiente o comprado).
- Usuario que lo añadió.
- Usuario que lo compró (si aplica).



Requisitos de información

Observaciones:

- Texto de la nota.
- Autor de la observación.
- Fecha de creación/modificación.



Requisitos de información

Pagos:

- ID de la lista.
- Nombre de la lista
- Participantes.
- Pagos
- Total
- Última actualización
- Quien actualizó



Requisitos de información

Precio total de la lista:

- Base para el cálculo del reparto de gastos en listas compartidas.



Requisitos funcionales

Gestión de usuarios

- Registrar un nuevo usuario mediante correo y contraseña.
- Iniciar sesión con credenciales válidas.
- Cerrar sesión.



Requisitos funcionales

Gestión de listas

- Crear una lista de compras.
- Eliminar una lista (solo el propietario).
- Compartir una lista mediante correo electrónico.
- Dejar de compartir una lista (eliminar usuario invitado).
- Ver listas propias y compartidas.
- Diferenciar visualmente entre listas individuales y compartidas.



Requisitos funcionales

Gestión de productos

- Añadir productos manualmente.
- Añadir productos mediante escaneo de código de barras (Android).
- Eliminar productos.
- Marcar productos como comprados.
- Mostrar en listas compartidas quién añadió y quién compró cada producto.



Requisitos funcionales

Gestión de observaciones

- Añadir observaciones a una lista.
- Modificar observaciones existentes.
- Eliminar observaciones.



Requisitos funcionales

Gestión económica

- Calcular el precio total de la lista.
- Calcular cuánto debe pagar cada usuario en listas compartidas.
- Marcar quién pagó y quién no.



Requisitos funcionales

Sincronización y plataforma

- Sincronización entre Android y Web.
- Acceso desde Android con todas las funciones (incluido escaneo) excepto eliminación de listas.
- Acceso desde Web con las mismas funciones además de eliminación de listas (excepto escaneo de código de barras).



Casos de uso

CU-01 Crear lista

Actor: Usuario registrado (propietario).

Descripción: El usuario crea una lista nueva, introduciendo un nombre identificativo.

CU-02 Compartir lista

Actor: Usuario propietario.

Descripción: El propietario introduce el correo de otro usuario y le otorga acceso a la lista.

CU-03 Añadir producto manualmente

Actor: Propietario o invitado.

Descripción: Se escribe el nombre de un producto y se añade a la lista seleccionada.

CU-04 Añadir producto por código de barras

Actor: Propietario o invitado (solo en Android).

Descripción: Se escanea un producto, y la app registra su información (nombre, marca y modelo).



Casos de uso

CU-05 Eliminar producto

Actor: Propietario o invitado.

Descripción: Se selecciona un producto de la lista y se elimina.

CU-06 Marcar producto como comprado

Actor: Propietario o invitado.

Descripción: Se marca un producto como adquirido; pasa a la sección “Comprados” y se registra qué usuario lo compró.

CU-07 Añadir observación

Actor: Propietario o invitado.

Descripción: El usuario escribe una nota dentro de la lista seleccionada.

CU-08 Modificar observación

Actor: Autor de la observación o propietario.

Descripción: Se edita el texto de una observación existente.



Casos de uso

CU-09 Calcular precio total de la lista

Actor: Propietario o invitado.

Descripción: El sistema suma los precios de todos los productos y muestra el gasto total estimado.

CU-11 Mostrar opciones de división de gasto

Actor: Propietario o invitado en lista compartida.

Descripción: A partir del precio total, el sistema reparte el gasto entre los usuarios asociados a la lista.

Actor: Propietario o invitado de lista compartida

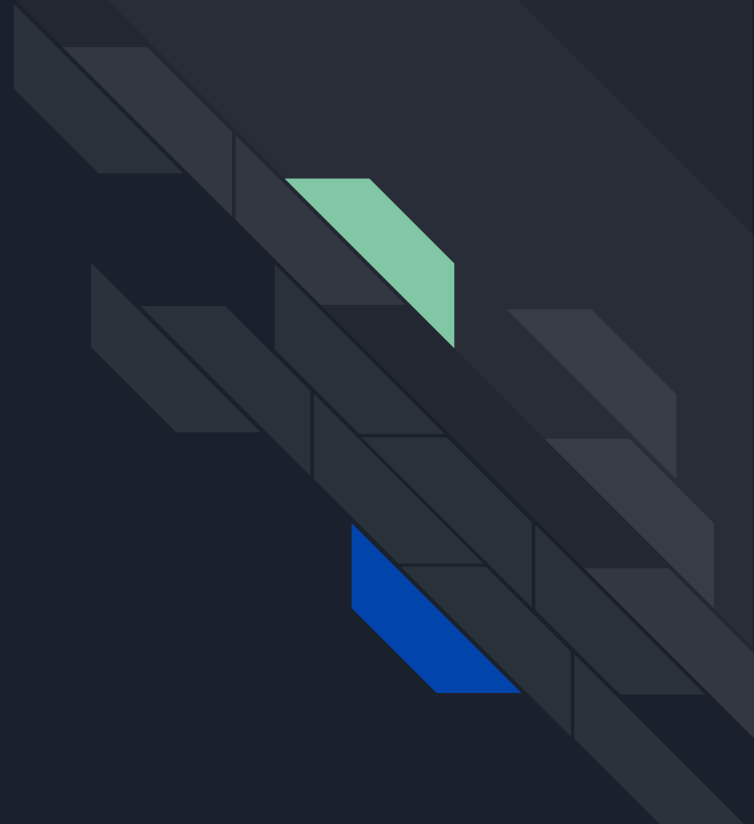
Descripción: En función de si es invitado o propietario tendrá una serie de conceptos

CU-12 Editar nombre de la lista

Actor: Propietario

Descripción: El actor puede modificar el nombre de la lista. El sistema muestra el nombre actual y permite editarlo. Una vez confirmado el cambio, el sistema actualiza y guarda el nuevo nombre de la lista.

Modelo de datos





Modelo conceptual

Se ha usado una base de datos de Firebase, base de datos NoSQL usando colecciones, documentos con JSON

- **Usuarios (*users*)**
Representan a las personas que usan la aplicación.
 - Cada usuario tiene un correo único, fecha de creación, último acceso y un identificador (*uid*).
 - Los usuarios pueden ser propietarios o invitados en las listas.
- **Listas (*lists*)**
Representan un conjunto de compras.
 - Cada lista tiene un nombre, un propietario, un conjunto de usuarios invitados y productos asociados.
 - Se guardan fechas relevantes (creación, finalización) y estados de la lista.
 - Se almacena el **precio total**.



Modelo conceptual

- **Productos (`items`)**
Representan artículos dentro de una lista.
 - Incluyen nombre, categoría, estado (pendiente/comprado), precio (si lo hay).
 - Guardan quién lo añadió y quién lo compró.
 - Están ligados a una lista mediante `listId`.
- **Observaciones (`listObservations`)**
Son notas o comentarios sobre una lista.
 - Tienen autor, fecha de última modificación y el contenido de la observación.
 - Se asocian a una lista.

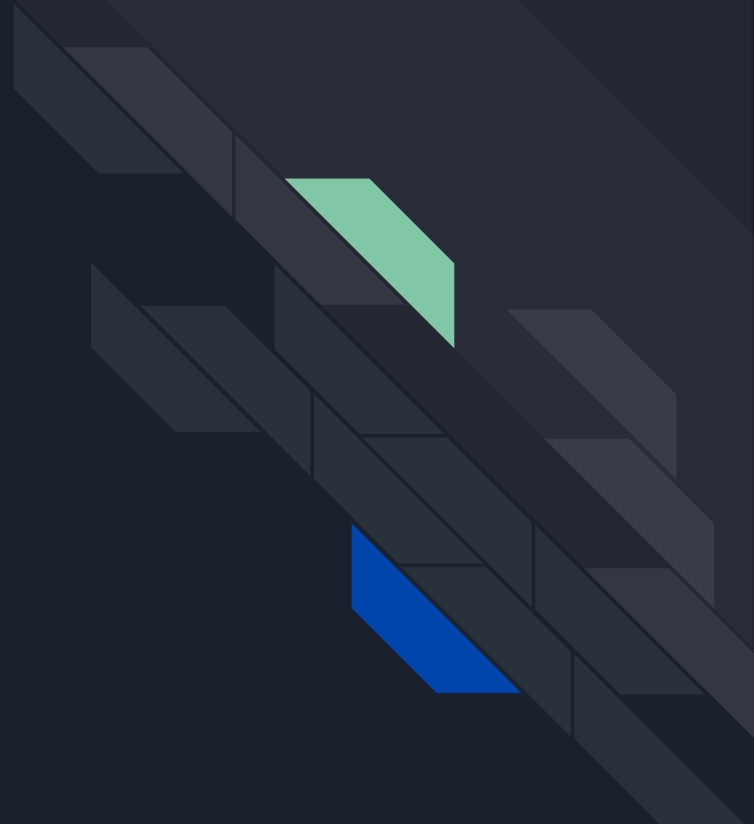


Modelo conceptual

- Pagos(**listPayments**)

Incluyen el nombre de la lista, ID, usuarios que han pagado, ultima actualización y quien lo actualizó

Descripción de entidades





Descripción de entidades

Usuarios (**users**)

- **uid** (string, clave primaria lógica)
- **email** (string, único)
- **createdAt** (timestamp)
- **lastLogin** (timestamp)
- **searchEmail** (string, para búsquedas rápidas)



Descripción de entidades

Listas (**lists**)

- **userId** (string, referencia al propietario en **users**)
- **ownerEmail** (string, redundancia para mostrar sin lookup)
- **name** (string)
- **createdAt** (timestamp)
- **finalizedAt** (timestamp, opcional)
- **finalizedBy** (string, correo de quien finalizó)
- **sharedWith** (array de correos)
- **purchaseInfo** (mapa con metadatos de compra)
- **totalPrice** (number, suma de todos los productos)



Descripción de entidades

Productos (**items**)

- **listId** (string, referencia a **lists**)
- **text** (string, nombre del producto)
- **category** (string, categoría general, ej. DAIRY)
- **completed** (boolean, comprado o no)
- **addedBy** (string, correo del usuario)
- **purchasedBy** (string, correo del comprador, opcional)
- **createdAt** (timestamp)
- **userId** (string, referencia a **users**)



Descripción de entidades

Observaciones (**listObservations**)

- **observations** (string)
- **updatedBy** (string, correo del autor/modificador)
- **updatedAt** (timestamp)



Descripción de entidades

Pagos(**listPayments**)

listId(string)

listName(string)

paymentParticipants(Array)

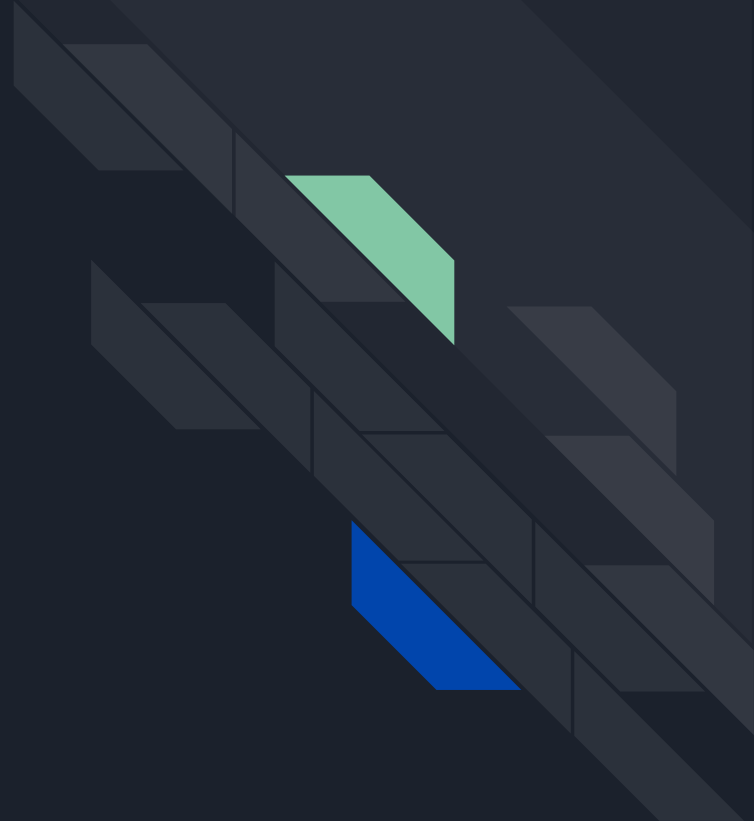
payments(Array)

totalAmount(int)

updatedAt(timestamp)

updatedBy(timestamp)

Descripción de interrelaciones





Descripción de interrelaciones

Un **usuario** puede tener **varias listas** (relación 1:N).

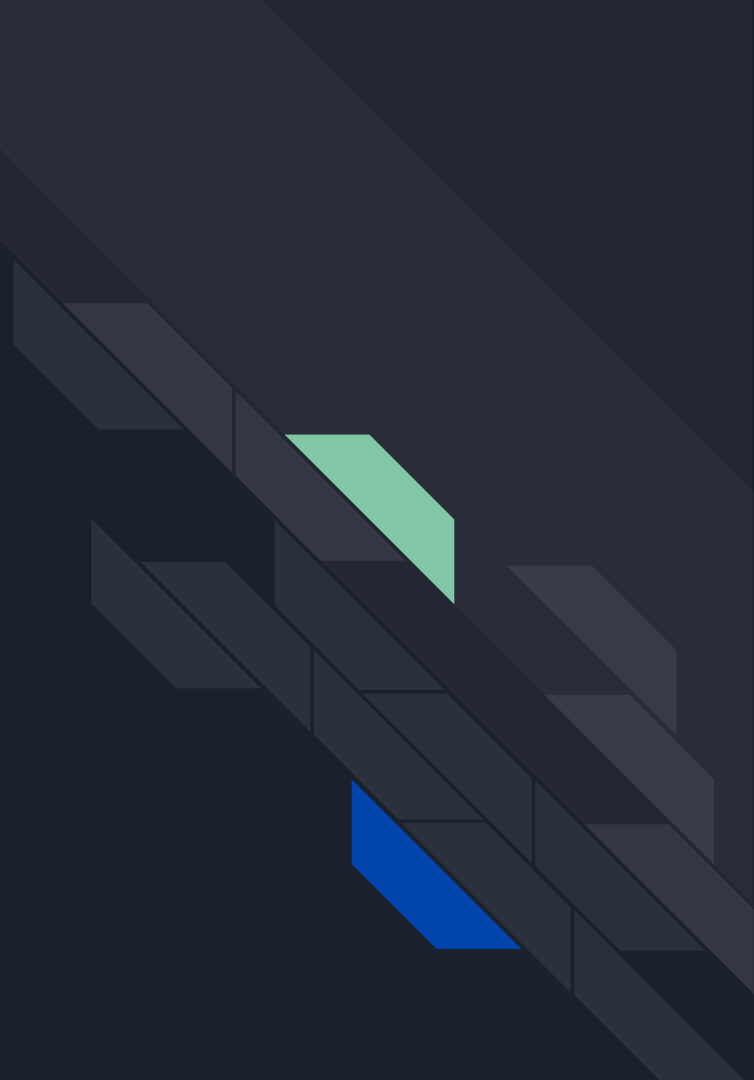
Una **lista** puede estar compartida con **varios usuarios** (relación N:M, implementada con **sharedWith**).

Una **lista** contiene **varios productos** (relación 1:N vía **listId**).

Una **lista** puede tener **varias observaciones** (relación 1:N).

Un **producto** pertenece a una sola lista, pero puede ser añadido y comprado por usuarios distintos.

Modelo lógico de datos






Modelo lógico de datos

users (colección)

└── {uid} (documento)

- email
- createdAt
- lastLogin
- searchEmail

Modelo lógico de datos



lists (colección)

└── {listId} (documento)

- name
- ownerEmail
- userId
- createdAt
- finalizedAt
- finalizedBy
- sharedWith [array]
- purchaseInfo {map}
- totalPrice
- expenseSplit {map}
- extraInfo



Modelo lógico de datos

items (colección)

└─ {itemId} (documento)

- listId
- text
- category
- completed
- addedBy
- purchasedBy
- createdAt
- userId



Modelo lógico de datos

listObservations (colección)

└── {observationId} (documento)

- listId
- observations
- updatedBy
- updatedAt



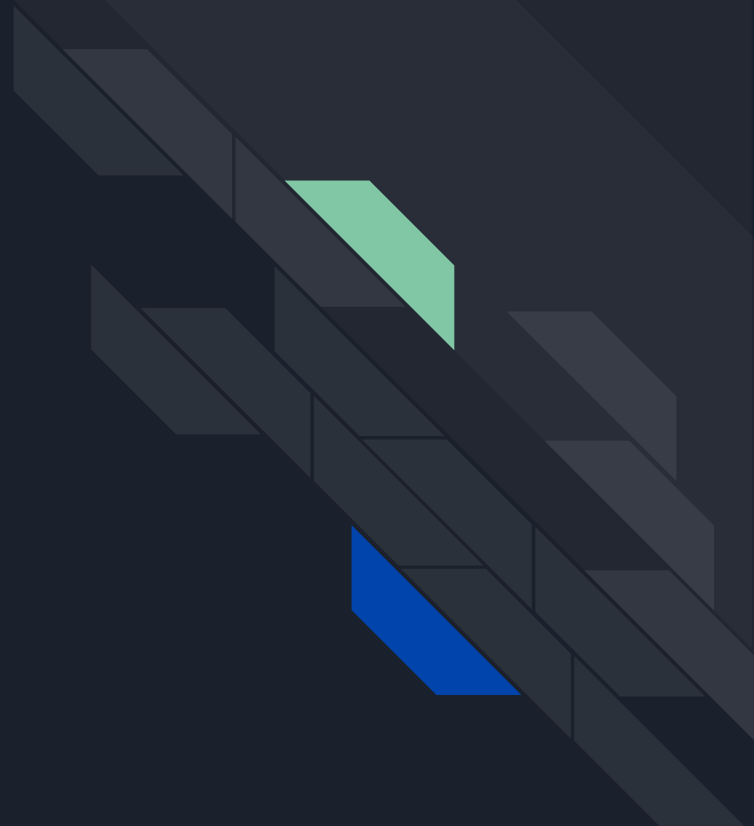
Modelo lógico de datos

listPayment(colección)

└── {itemId} (documento)

- listId
- listName
- paymentParticipants
- payments
- totalAmount
- updatedAt
- updatedBy

Consultas





Consultas

Obtener todas las listas de un usuario (propietario o compartidas):

```
db.collection("lists")  
  .where("ownerEmail", "==", userEmail)  
  .get()
```

```
db.collection("lists")  
  .where("sharedWith", "array-contains", userEmail)  
  .get()
```



Consultas

Obtener productos de una lista:

```
db.collection("items")  
  .where("listId", "==", listId)  
  .get()
```



Consultas

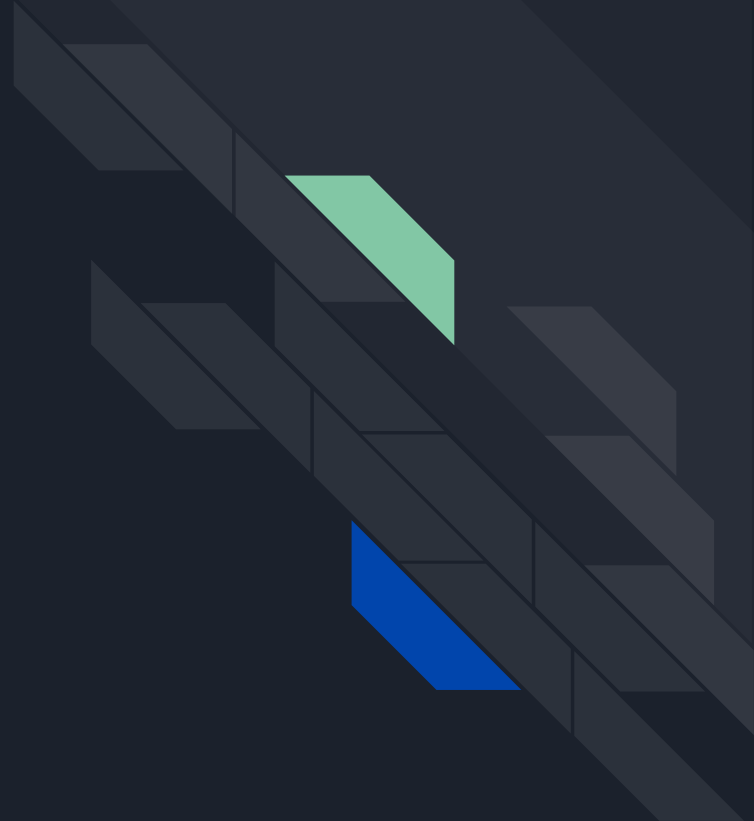
Mostrar observaciones de una lista:

```
db.collection("listObservations")  
  .where("listId", "==", listId)  
  .orderBy("updatedAt", "desc")  
  .get()
```

Mostrar pagos de una lista

```
db.collection("listPayments")  
  .where("listId", "==", listId)  
  .orderBy("updatedAt", "desc")  
  .get()
```

Interfaz de usuario



BuyNote

Texto en función de si es
registro o inicio de sesión

Email

Contraseña

Iniciar sesión/Registrarse

Cambio a inverso

Consejo al usuario



Listas

Nueva Lista



Lista X

Nombre de la lista



Precio total:
Observaciones

Productos

Nuevo Producto



Categoría 1



Producto 1



Categoría 2



Producto 2



BuyNote

Texto en función de si es
registro o inicio de sesión

Email

Contraseña

Iniciar sesión/Registrarse

Cambio a inverso

Consejo al usuario



Listas

Compartidas

L1



L1



L1



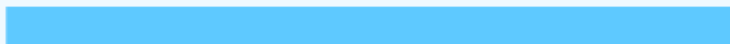
L1



Crear nueva lista

Precio total:

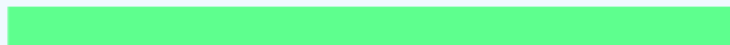
Productos



P1



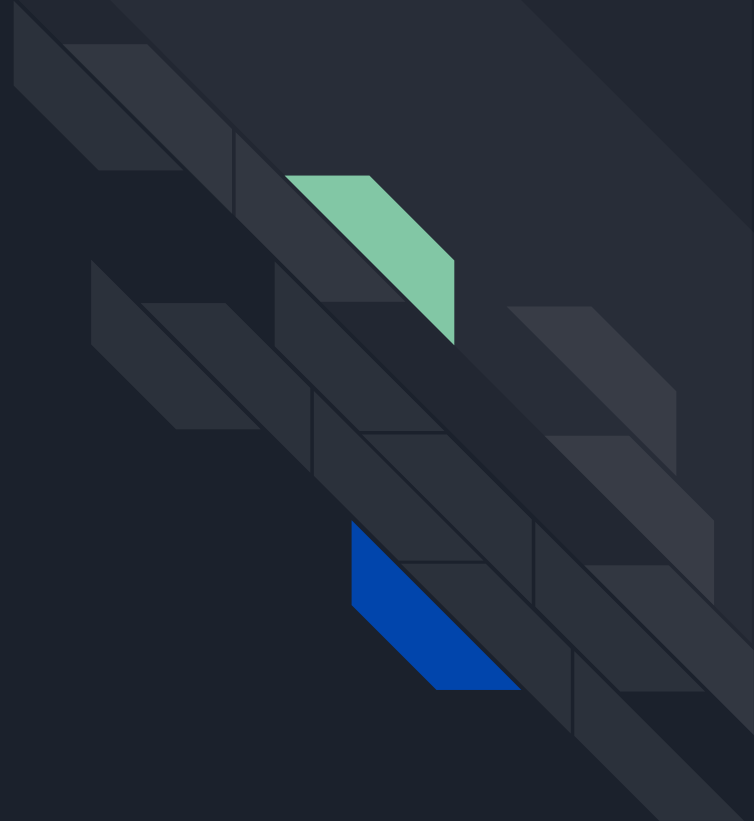
P1

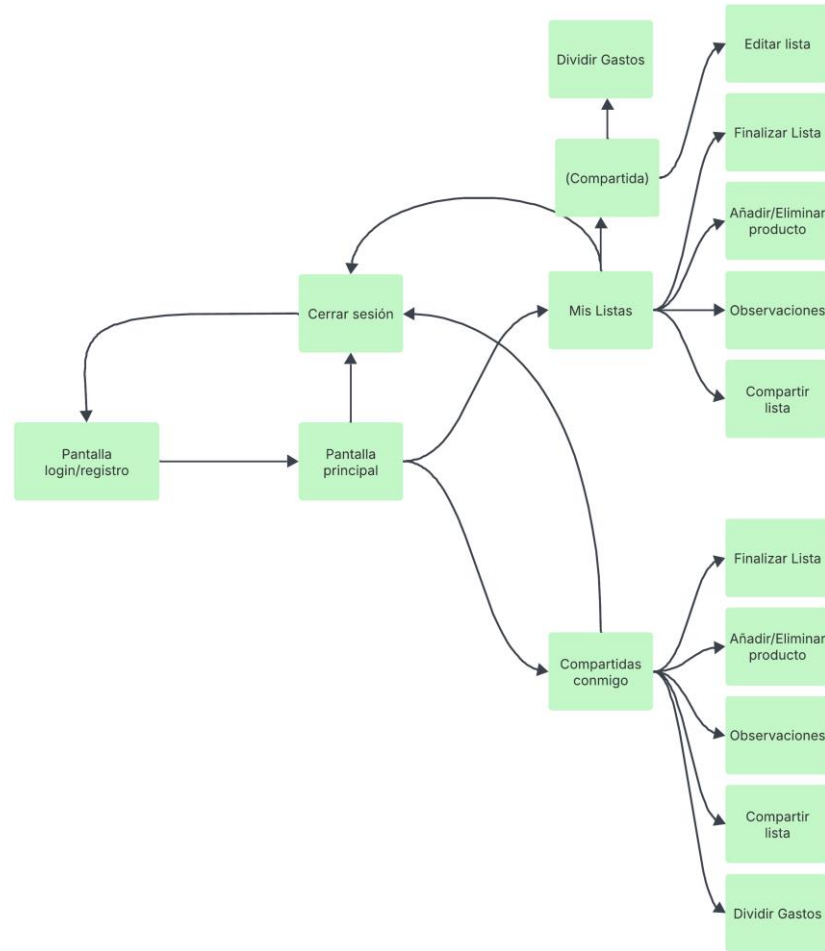


P1

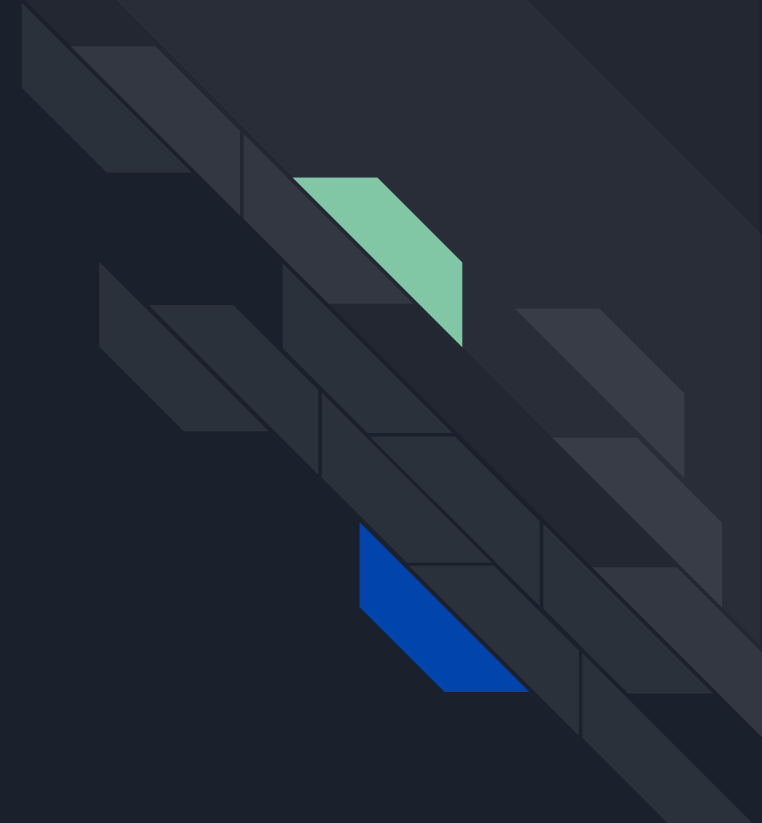


Mapa de navegación





Desarrollo del proyecto del sistema





Desarrollo del proyecto del sistema

Fase 1

Funcionalidades

Android:

- Registro e inicio de sesión de usuarios (correo y contraseña).
- Crear listas de compras (listas individuales).
- Ver listas propias (menú principal).
- Añadir productos manualmente a una lista.
- Marcar productos como comprados.
- Eliminar productos de la lista.

Web:

- Registro e inicio de sesión de usuarios (correo y contraseña).
- Crear listas de compras.
- Eliminar listas.
- Añadir, eliminar y compartir productos dentro de las listas.



Desarrollo del proyecto del sistema

Plan de pruebas

Android

- **Prueba de registro:** crear un usuario nuevo; comprobar que se guarda en Firestore y no permite duplicados.
- **Prueba de inicio de sesión:** iniciar con usuario válido y no registrado; verificar mensajes de error.
- **Prueba de creación de lista:** crear lista con nombre; comprobar que aparece en el menú principal.
- **Prueba de añadir productos:** añadir productos manualmente; verificar que se guardan con `listId` correcto.
- **Prueba de marcar productos como comprados:** marcar un producto y comprobar que cambia su estado a `completed = true` y se mueve a “Comprados”.
- **Prueba de eliminación de productos:** eliminar producto y verificar que desaparece tanto de la interfaz como de Firestore.



Desarrollo del proyecto del sistema

Plan de pruebas

Web

- **Prueba de registro e inicio de sesión:** registrar y acceder con usuario válido; validar errores en datos incorrectos.
- **Prueba de creación de lista:** crear lista y verificar que se muestra correctamente.
- **Prueba de eliminación de lista:** eliminar lista y comprobar su eliminación en Firestore.
- **Prueba de productos:** añadir y eliminar productos, comprobando su persistencia en la base de datos.
- **Prueba de compartición:** compartir lista con otro usuario y verificar que tiene acceso a ella.



Desarrollo del proyecto del sistema

Fase 2:

Funcionalidades

1. Añadir productos mediante código de barras (Android).
2. Compartir listas con otros usuarios mediante correo electrónico.
3. Añadir, modificar y eliminar observaciones.
4. Calcular precio total de la lista y cuánto paga cada usuario.
5. Diferenciación visual de listas (individuales vs compartidas).
6. Sincronización en tiempo real entre Android y Web.
7. Finalizar listas y registrar quién finalizó la compra.
8. Editar nombres de las listas

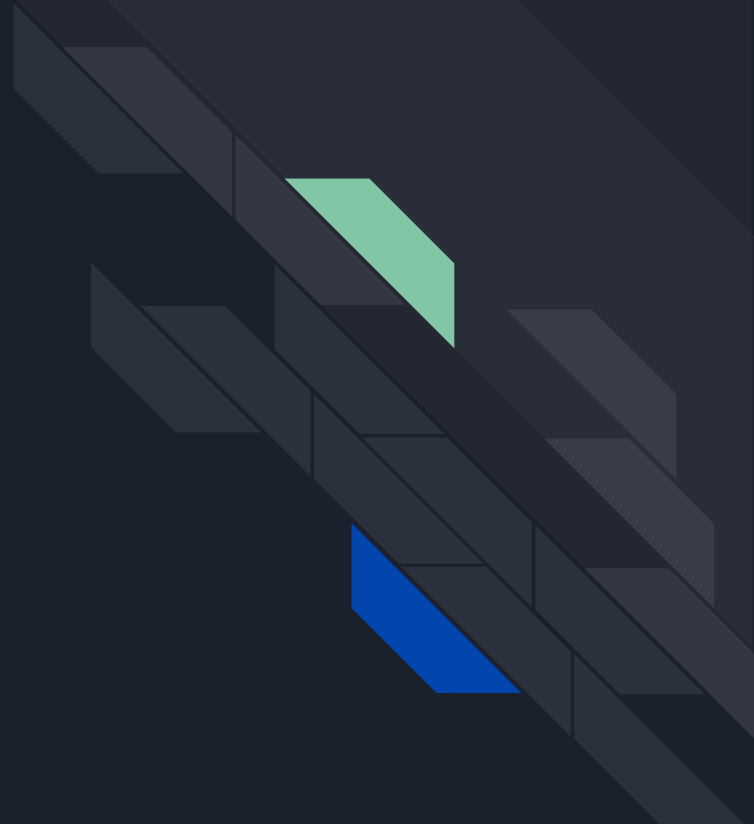


Desarrollo del proyecto del sistema

Plan de pruebas

- **Prueba de observaciones:** crear, modificar y eliminar observaciones; comprobar autor y fecha.
- **Prueba de cálculo de gastos:** introducir precios y verificar el total y el reparto.
- **Prueba de diferenciación visual:** confirmar que listas compartidas se muestran con diseño distinto.
- **Prueba de sincronización:** modificar datos desde Android y comprobar reflejo inmediato en Web.
- **Prueba de finalización:** finalizar lista y verificar campos `finalizedAt` y `finalizedBy` en Firestore.
- **Prueba de edición:** Editar el nombre de una lista y comprobar que se modifica

Información para el
cliente





Información para el cliente

Requisitos físicos y de hardware

Para Android:

- Dispositivo con sistema operativo Android 8.0 (Oreo) o superior.
- Memoria RAM mínima: 2 GB.
- Espacio de almacenamiento disponible: 50 MB aprox.
- Cámara (opcional, solo si quiere usar la función de escaneo de códigos de barras).
- Conexión a Internet estable para sincronización y listas.

Para Web:

- Ordenador, portátil o tablet con navegador moderno: Chrome, Firefox, Edge o Safari actualizado.
- Conexión a Internet estable para sincronización y listas.



Información para el cliente

Requisitos software

Para Android:

- App instalada desde Google Play o mediante APK proporcionado.
- Sistema operativo Android 8.0 o superior.
- Permisos de la app: cámara (solo si se va a usar escaneo de códigos de barras).

Para Web:

- Navegador actualizado: Chrome, Firefox, Edge o Safari.
- Capacidad de conectarse al dominio de la app.

Común a ambas plataformas:

- Cuenta de usuario con correo electrónico y contraseña.
- Conexión a Internet para ver y sincronizar listas



Información para el cliente

Manual de instalación, configuración y administración

Instalación en Android:

1. Descargar la app desde Google Play o instalar el APK proporcionado.
2. Conceder permisos de cámara si se desea usar el escaneo de códigos de barras.
3. Abrir la app e iniciar sesión o registrarse con correo y contraseña.
4. La app se sincroniza automáticamente con la nube (Firestore).



Información para el cliente

Manual de instalación, configuración y administración

Instalación en Web:

1. Acceder al dominio de la app.
2. Registrarse o iniciar sesión con correo y contraseña.
3. Todas las listas y productos se sincronizan automáticamente.



Información para el cliente

Manual de instalación, configuración y administración

Configuración y administración de listas:

- Crear listas individuales o compartirlas con otros usuarios mediante correo electrónico.
- Añadir, modificar o eliminar productos y observaciones.
- Añadir precios a productos y ver el total de la lista.
- Calcular automáticamente cuánto debe pagar cada usuario en listas compartidas.
- Finalizar listas y administrar permisos de usuarios invitados.



Información para el cliente

Manual de usuario

1. Registro e inicio de sesión:

- Crear una cuenta con correo y contraseña.
- Iniciar sesión con usuario registrado.

2. Crear lista de compras:

- Pulsar el botón “Crear lista”, introducir un nombre y guardar.

3. Editar lista de compras:

- Pulsar el botón del lápiz,, modificar el nombre y guardar.

3. Añadir productos:

- **Manual:** escribir el nombre del producto y guardarlo.
- **Android (opcional):** usar el escáner de códigos de barras para añadir automáticamente nombre, marca y modelo.



Información para el cliente

4. Gestionar productos:

- Marcar productos como comprados.
- Eliminar productos.
- Ver productos comprados al final de la lista para mantener claridad.
- Ver los precios de los productos

5. Observaciones:

- Añadir notas sobre la lista o productos.
- Modificar o eliminar observaciones si es necesario.

6. Compartir listas:

- Introducir el correo del usuario invitado y enviar invitación.
- Ver quién ha añadido y quién ha comprado cada producto.



Información para el cliente

7. Gestión económica:

- Añadir precios a productos.
- Consultar **total de la lista**.
- Calcular automáticamente cuánto debe pagar cada usuario en listas compartidas.
- Finalizar automáticamente o manualmente la lista de la compra

8. Sincronización y finalización de listas:

- Todos los cambios se reflejan en tiempo real en Android y Web.
- Finalizar lista cuando se completa la compra.



Bibliografía y fuentes de información

[Chatgpt](#), [Deepseek](#) → Cuestiones de código

[Figma](#) → Creación de pantallas del software

[LucidChart](#) → Creación de mapa de navegación