



# BEHOME

## DESARROLLO DE UNA APLICACIÓN MÓVIL PARA LA GESTIÓN DEL HOGAR

Desarrollo de Aplicaciones Multiplataforma

13/06/2024

Autor: Sara Alejo Millán

Tutores: Israel Francisco Gimeno y Luis Hueso Ibañez



**IES Santiago Hernández**

Zaragoza, España

# ÍNDICE

<b>1. Análisis y descripción.....</b>	<b>1</b>
1.1. Descripción del producto.....	1
1.2. Descripción del proyecto.....	2
1.3. Justificación del proyecto y tecnologías.....	5
1.3.1. Estudio de mercado.....	5
1.3.2. Justificación y Oportunidad.....	6
1.3.3. Análisis DAFO y CAME.....	8
1.3.4. Tecnologías.....	11
1.4. Requisitos funcionales y no funcionales.....	12
1.4.1. Requisitos funcionales:.....	12
1.4.2. Requisitos No Funcionales.....	14
1.5. Planificación del proyecto.....	15
1.5.1. Fases.....	15
1.5.2. Diagrama de Gantt.....	17
<b>2. Análisis y diseño.....</b>	<b>18</b>
2.1. Diagrama de Casos de Uso.....	18
2.2. Diagrama de Clases.....	19
2.3. Diagrama de Secuencia de la aplicación.....	20
2.4. Modelos de datos.....	21
2.4.1. Modelo lógico de datos.....	21
2.4.1.1. Diagrama Entidad-Relación.....	22
2.4.1.2. Diagrama Relacional.....	23
2.5. Desarrollo de Interfaces.....	24
<b>3. Configuración y desarrollo del software.....</b>	<b>28</b>
3.1. Implementación y pruebas.....	28
3.2. Memoria económica.....	36
<b>4. Conclusión.....</b>	<b>39</b>
<b>5. Bibliografía.....</b>	<b>40</b>

## ÍNDICE DE FIGURAS

<b>Figura 1. Icono de la Aplicación.....</b>	<b>2</b>
<b>Figura 2. Dibujo Simple de las Interfaces .....</b>	<b>4</b>
<b>Figura 3. Análisis DAFO y CAME.....</b>	<b>8</b>
<b>Figura 4. Diagrama de Gantt.....</b>	<b>17</b>
<b>Figura 5. Diagrama de Casos de Uso.....</b>	<b>18</b>
<b>Figura 6. Diagrama de Clases.....</b>	<b>19</b>
<b>Figura 7. Diagrama de Secuencia de la Aplicación.....</b>	<b>20</b>
<b>Figura 8. Diagrama Entidad-Relación.....</b>	<b>22</b>
<b>Figura 9. Diagrama Relacional .....</b>	<b>23</b>
<b>Figura 10. Interfaz de Inicio de Sesión y Registro.....</b>	<b>24</b>
<b>Figura 11. Interfaz de Crear un Piso y Unirse a un Piso.....</b>	<b>24</b>
<b>Figura 12. Interfaz Principal de Resumen.....</b>	<b>25</b>
<b>Figura 13. Interfaz de Tareas y Crear Tarea.....</b>	<b>26</b>
<b>Figura 14. Interfaz de Listas, Crear Lista y Añadir Producto.....</b>	<b>26</b>
<b>Figura 15. Interfaz de Calendario.....</b>	<b>27</b>
<b>Figura 16. Interfaz de Perfil.....</b>	<b>27</b>
<b>Figura 17. Inicio de Sesión: Email incorrecto, Contraseña incorrecta, Credenciales correctas y Pantalla principal de la aplicación.....</b>	<b>32</b>
<b>Figura 18. Pantallas de registro incorrecto .....</b>	<b>32</b>
<b>Figura 19. Registro correcto y creación de un piso.....</b>	<b>33</b>
<b>Figura 20. Vista del Calendario y creación de un Evento .....</b>	<b>33</b>
<b>Figura 21. Creación de una lista y sus productos.....</b>	<b>34</b>
<b>Figura 22. Vista del Perfil y Cierre de Sesión.....</b>	<b>34</b>
<b>Figura 23. Tabla de Costes Fijos.....</b>	<b>37</b>
<b>Figura 24. Tabla de Recursos Materiales.....</b>	<b>37</b>

# 1. Análisis y descripción

## 1.1. Descripción del producto

BeHome es una aplicación móvil diseñada para facilitar la gestión y organización del hogar. Esta aplicación busca simplificar las tareas diarias, mejorar la comunicación entre los miembros de la familia o compañeros de piso, y proporcionar una experiencia más ordenada y eficiente en la administración de las responsabilidades del hogar.

Los usuarios pueden crear y asignar tareas de manera fácil y rápida. La aplicación permite programar y recordar las tareas del día, la semana y el mes. Con una interfaz intuitiva, los usuarios pueden marcar las tareas completadas y visualizar el progreso de cada miembro del hogar, fomentando un ambiente de colaboración y organización.

Además, ofrece la posibilidad de gestionar listas de la compra de manera eficiente. Los usuarios pueden crear y compartir listas de productos, añadir ítems con facilidad y marcar los productos adquiridos. Esta funcionalidad no solo ayuda a mantener el hogar abastecido, sino que también facilita la planificación y el control de gastos en el supermercado.

## 1.2. Descripción del proyecto

BeHome es una aplicación diseñada para mejorar la convivencia y la gestión del hogar compartido entre compañeros de piso. A continuación, se detallan los aspectos clave:

### **Contexto y Justificación:**

Esta idea surge como respuesta a la necesidad de simplificar la vida en hogares compartidos. La convivencia puede ser complicada, y esta aplicación busca facilitarla, al ofrecer herramientas para la gestión de gastos, asignación de tareas y comunicación efectiva.

Este sería el icono de la aplicación: una casa verde con un calendario dentro. Esta combinación simboliza la gestión del hogar y la organización de eventos y actividades dentro de él.



*Figura 1. Icono de la Aplicación*

Los colores corporativos de BeHome son principalmente el verde, morado y beige. Estos colores han sido elegidos por su significado y las diferentes emociones que pueden transmitir:

- **Verde** (#5AA37D)
- **Verde Claro** (#C2EABE)
- **Morado** (#85579E)
- **Beige** (#FBF3E5)

### **Objetivos Generales:**

- Facilitar la vida diaria de los compañeros de piso al proporcionar una plataforma centralizada.
- Mejorar la organización y la toma de decisiones en el hogar.

### **Funcionalidades Clave:**

- Gestión de Gastos:
  - BeHome permite registrar y dividir equitativamente los costos de compras, facturas y otros gastos relacionados con el hogar.
- Asignación de Tareas:
  - Los usuarios pueden asignar y dar seguimiento a tareas domésticas, asegurando una distribución justa de responsabilidades.
  - Se pueden agregar notificaciones para recordar las tareas semanales.
- Comunicación Efectiva:
  - La aplicación ofrece un sistema de mensajería interna para coordinar eventos, cambios en la rutina y otros temas relacionados con la casa.
- Lista de Compras Compartida:
  - Los compañeros de piso pueden colaborar en una lista de compras actualizada en tiempo real.

Dibujo simple de las diferentes interfaces:

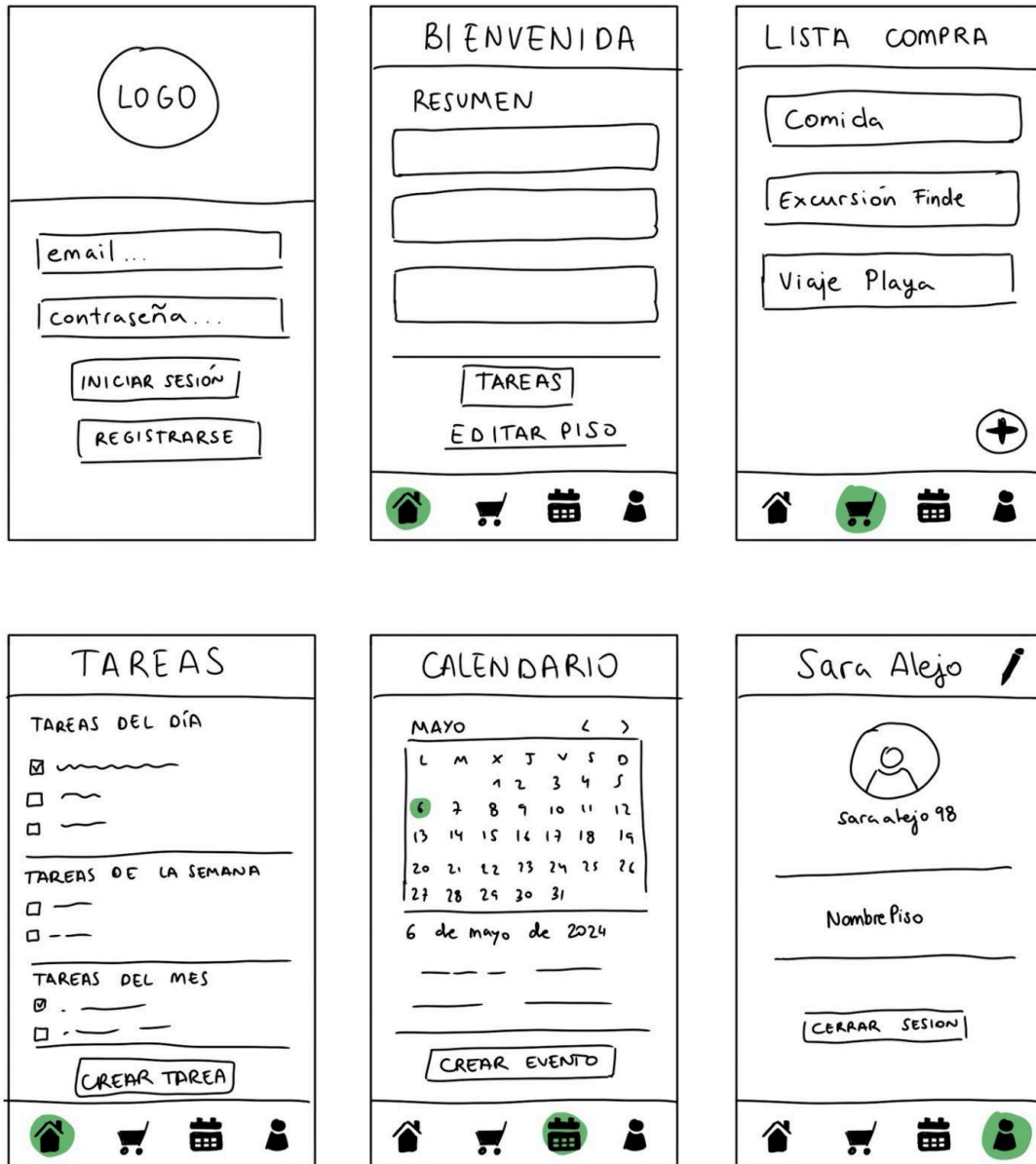


Figura 2. Dibujo Simple de las Interfaces

## 1.3. Justificación del proyecto y tecnologías

### 1.3.1. Estudio de mercado

#### **Flatify:**

- Fortalezas
  - Tiene un diseño minimalista y moderno que puede atraer a usuarios que prefieren una interfaz limpia y simple
  - La aplicación combina la gestión de tareas del hogar con listas de la compra, lo que puede ser conveniente para los usuarios que desean unificar estas funciones en una sola plataforma.
  - Permite compartir listas de compras y tareas del hogar con otros usuarios, facilitando la colaboración en el hogar.
- Debilidades
  - Carece de algunas características avanzadas como recordatorios específicos para tareas, integración con calendarios u opciones de personalización.
  - No está disponible en todos los idiomas.

#### **Listonic:**

- Fortalezas:
  - Está especializada en listas de la compra, ofreciendo funciones avanzadas como reconocimiento de voz, categorización de productos y sincronización entre dispositivos.
  - Tiene la capacidad de agregar elementos a la lista de la compra mediante comandos de voz puede ser una característica atractiva para los usuarios ocupados.
- Debilidades:
  - Se centra exclusivamente en listas de la compra y puede carecer de características para la gestión de tareas del hogar o calendario compartido.
  - Interfaz muy simple, con pocas opciones de personalización.



### **Todoist:**

- **Fortalezas:**
  - Ofrece una amplia gama de características avanzadas como etiquetas, prioridades, recordatorios, sub-tareas y colaboración en proyectos compartidos.
  - Se integra fácilmente con calendarios como Google Calendar y aplicaciones de productividad como Slack y Trello, proporcionando una experiencia de gestión de tareas más completa.
  - Está disponible en múltiples plataformas, incluyendo Android, iOS, Windows, macOS y web, lo que permite a los usuarios acceder a sus tareas desde cualquier dispositivo.
- **Debilidades:**
  - Tantas funcionalidades pueden llegar a resultar abrumadoras para nuevos usuarios.
  - Algunas características avanzadas de Todoist, como etiquetas de proyecto y recordatorios basados en ubicación, requieren una suscripción premium, lo que puede ser un obstáculo para algunos usuarios.

### **1.3.2. Justificación y Oportunidad**

#### **Público al que va destinado:**

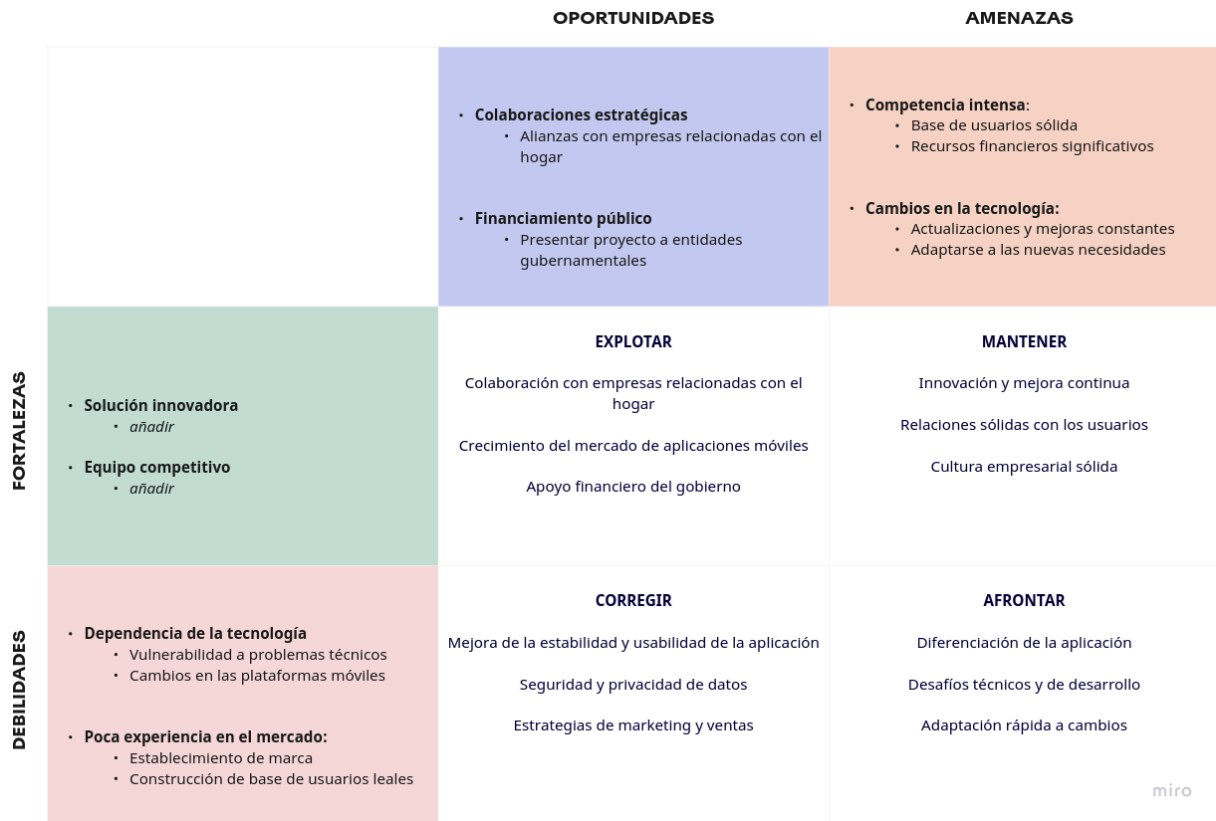
BeHome está dirigido a jóvenes y adultos que comparten un espacio de vida, ya sea en apartamentos, casas o residencias estudiantiles. Es ideal para aquellos que buscan una solución eficiente para la gestión de tareas y gastos compartidos, así como una comunicación efectiva entre compañeros de piso.

#### **Posibilidades que ofrece:**

- **Transparencia Financiera:** Permite a los usuarios tener una visión clara de los gastos compartidos y las contribuciones individuales, fomentando la transparencia y evitando malentendidos.

- **Ahorro de Tiempo:** Simplifica la planificación y coordinación diaria, ahorrando tiempo a los usuarios al evitar la necesidad de comunicarse constantemente sobre las responsabilidades del hogar.
- **Fomento de la Colaboración:** Facilita la colaboración entre compañeros de piso al proporcionar herramientas intuitivas para la gestión de tareas y la comunicación efectiva, mejorando así la convivencia.
- **Organización Eficiente:** Ayuda a mantener el hogar organizado y en armonía al proporcionar un lugar centralizado para todas las actividades relacionadas con la vida compartida.

### 1.3.3. Análisis DAFO y CAME



*Figura 3. Análisis DAFO y CAME*

Debilidades:

- **Dependencia de la tecnología:** La empresa puede ser vulnerable a problemas técnicos y cambios en las plataformas móviles, lo que podría afectar la experiencia del usuario y la eficacia de la aplicación.
- **Poca experiencia en el mercado:** BeHome es una empresa nueva y puede enfrentar desafíos para establecer su marca y construir una base de usuarios leales debido a la competencia en el mercado de aplicaciones móviles relacionadas con la gestión del hogar.

#### Amenazas:

- **Competencia intensa:** Existen otras aplicaciones de gestión del hogar establecidas en el mercado que pueden tener una base de usuarios sólida y recursos financieros significativos, lo que podría dificultar la diferenciación y el crecimiento de BeHome.
- **Cambios en la tecnología:** Los avances tecnológicos rápidos pueden hacer que la aplicación de BeHome se vuelva obsoleta rápidamente si no se mantienen actualizaciones y mejoras constantes, lo que podría afectar su relevancia y utilidad para los usuarios.

#### Fortalezas:

- **Solución innovadora:** BeHome ofrece una solución innovadora para la gestión del hogar compartido, lo que puede generar interés entre los consumidores que buscan una forma más eficiente de organizar sus vidas domésticas.
- **Equipo competitivo:** La empresa cuenta con un equipo de profesionales con experiencia en desarrollo de aplicaciones móviles, diseño de interfaz de usuario y marketing digital, lo que puede ayudar a impulsar el éxito del proyecto.

#### Oportunidades:

- **Colaboraciones estratégicas:** BeHome puede explorar oportunidades para asociarse con empresas relacionadas con el hogar, como supermercados o servicios de limpieza, para ampliar su alcance y generar ingresos adicionales a través de acuerdos de marketing o publicidad.
- **Financiamiento público:** Presentar el proyecto a entidades gubernamentales puede resultar en financiamiento o subvenciones para promover la conciliación familiar y mejorar la eficiencia en la gestión del hogar, lo que podría proporcionar un impulso significativo a BeHome.

Corregir:

- Mejorar la estabilidad y la usabilidad de la aplicación para garantizar una experiencia de usuario óptima.
- Abordar cualquier problema de seguridad o privacidad de datos para cumplir con las regulaciones vigentes y aumentar la confianza de los usuarios.
- Capacitar al equipo en estrategias de marketing y ventas para aumentar la visibilidad de la marca y adquirir nuevos usuarios.

Afrontar:

- Enfrentar la competencia mediante la diferenciación de la aplicación a través de características únicas y un enfoque en la experiencia del usuario.
- Afrontar los desafíos técnicos y de desarrollo mediante la implementación de un proceso robusto de control de calidad y pruebas de usuario.
- Estar preparado para adaptarse rápidamente a los cambios en el mercado y la tecnología mediante actualizaciones regulares y mejoras en la aplicación.

Mantener:

- Mantener el enfoque en la innovación y la mejora continua de la aplicación para mantenerse relevante en un mercado en constante evolución.
- Mantener relaciones sólidas con los usuarios existentes y fomentar la retroalimentación para mejorar constantemente la aplicación según las necesidades del mercado.
- Mantener una cultura empresarial sólida que fomente la colaboración, la creatividad y el compromiso del equipo.

Explotar:

- Explotar las oportunidades de colaboración con empresas relacionadas con el hogar para aumentar los ingresos a través de acuerdos de marketing y publicidad.
- Explotar el potencial de crecimiento del mercado de aplicaciones móviles mediante estrategias de expansión y marketing efectivas.
- Explotar el respaldo y el apoyo financiero del gobierno para financiar iniciativas que promuevan la conciliación familiar y la eficiencia en la gestión del hogar.

#### 1.3.4. Tecnologías

- Sistema Operativo
  - Windows 11 Pro para ordenador
  - Android para dispositivos móviles (Emulador del propio IDE)
- Lenguaje de programación
  - Java 8
  - XML para Android Studio
  - SQL para BBDD
- Hardware
  - Ordenador portátil
  - Dispositivo móvil Android para pruebas de compatibilidad
- Software
  - Documentación
    - Microsoft Visual Studio Code 1.89.1
  - Entorno de Desarrollo
    - Android Studio 2022.3

- Control de Versiones
  - GitHub
- Base de datos
  - XAMPP 8.2.12-0
- Herramientas de diseño
  - Miro para diseño de interfaces y diagramas
  - Canva para diagramas y esquemas
  - Material Design
  - App Procreate 5.3.7

## 1.4. Requisitos funcionales y no funcionales

### 1.4.1. Requisitos funcionales:

#### **1. Registro de Usuarios:**

- a. Los usuarios pueden registrarse en la aplicación utilizando su dirección de correo electrónico o iniciar sesión con una cuenta de Google/Facebook.

#### **2. Gestión de Tareas del Hogar:**

- a. Los usuarios pueden crear, editar y eliminar tareas del hogar.
- b. Pueden asignar tareas a miembros específicos del hogar.
- c. La aplicación debe permitir establecer recordatorios para las tareas.

#### **3. Listas de la Compra:**

- a. Los usuarios pueden crear listas de la compra para diferentes tiendas o categorías.
- b. Pueden agregar, eliminar y marcar elementos como comprados.

- c. La aplicación puede ofrecer sugerencias de productos basadas en historiales de compras anteriores.

#### **4. Calendario Compartido:**

- a. Los usuarios pueden ver y compartir un calendario compartido con otros miembros del hogar.
- b. Pueden agregar eventos, recordatorios y citas.
- c. Se pueden establecer recordatorios para eventos importantes.

#### **5. Notificaciones:**

- a. La aplicación enviará notificaciones para recordar a los usuarios sobre tareas pendientes, eventos próximos y elementos de la lista de la compra.

#### **6. Gestión de Estancias:**

- a. Los usuarios pueden definir estancias (por ejemplo, salón, cocina) y asignarlas a pisos específicos.

#### **7. Perfil de Usuario:**

- a. Los usuarios pueden actualizar su información personal, como nombre, correo electrónico y preferencias de notificación.
- b. Los usuarios pueden ver y gestionar las listas de la compra y tareas que tienen asignadas.

#### **8. Historial de Actividades:**

- a. Los usuarios pueden ver un historial de tareas completadas y elementos de la lista de la compra marcados como comprados.



## 1.4.2. Requisitos No Funcionales

### **1. Seguridad y Privacidad:**

- a. Los datos de los usuarios deben estar seguros y protegidos mediante medidas de cifrado y autenticación.
- b. La aplicación debe cumplir con las regulaciones de privacidad de datos.

### **2. Rendimiento:**

- a. La aplicación debe ser rápida y receptiva, incluso en dispositivos móviles más antiguos.
- b. Debe tener tiempos de carga rápidos y una interfaz de usuario fluida.

### **3. Compatibilidad:**

- a. La aplicación debe ser compatible con una amplia gama de dispositivos Android y versiones del sistema operativo.

### **4. Usabilidad:**

- a. La interfaz de usuario debe ser intuitiva y fácil de usar, con un diseño limpio y organizado.
- b. Debe haber soporte para múltiples idiomas y ajustes de accesibilidad.

### **5. Escalabilidad:**

- a. La aplicación debe ser capaz de manejar un crecimiento en el número de usuarios y datos sin comprometer el rendimiento.

### **6. Disponibilidad:**

- a. Debe estar operativa y accesible durante la mayor parte del tiempo. Esto implica minimizar el tiempo de inactividad y garantizar que la aplicación pueda ser utilizada de manera confiable.

## **7. Mantenimiento:**

- a. Facilidad para actualizar y corregir errores.

## **1.5. Planificación del proyecto**

### **1.5.1. Fases**

- **Fase 1: Planificación Inicial**

- Identificación
  - Especificar requisitos funcionales y no funcionales
  - Documentación
- Análisis de viabilidad
  - Análisis técnico y económico
  - Estudio de mercado
- Plan de Proyecto
  - Plan de trabajo (ganttt)
  - Identificar y asignar recursos

- **Fase 2: Diseño del Sistema**

- Diseño de Arquitectura
  - Definir arquitectura cliente-servidor
  - Elegir tecnologías y herramientas necesarias
- Modelado de Datos
  - Diseñar modelo entidad-relación
  - Crear diagrama de clases, casos de uso y secuencia
- Prototipado
  - Desarrollar interfaces de usuario

- **Fase 3: Desarrollo del Sistema**

- Desarrollo Back
  - Configurar servidor XAMPP
  - Desarrollar lógica de negocio
- Desarrollo Front

- Desarrollar interfaz en Android Studio
- Integración y pruebas
  - Realizar pruebas unitarias y generales
  - Corregir errores
  - Optimizar código
- **Fase 4: Implementación y Despliegue**
  - Preparación del entorno
    - Configurar BBDD
  - Migración de datos
    - Importar datos iniciales de prueba
- **Fase 5: Puesta en Marcha y Formación**
  - Formación de Usuarios:
    - Crear materiales de formación (manuales o tutoriales).

### 1.5.2. Diagrama de Gantt

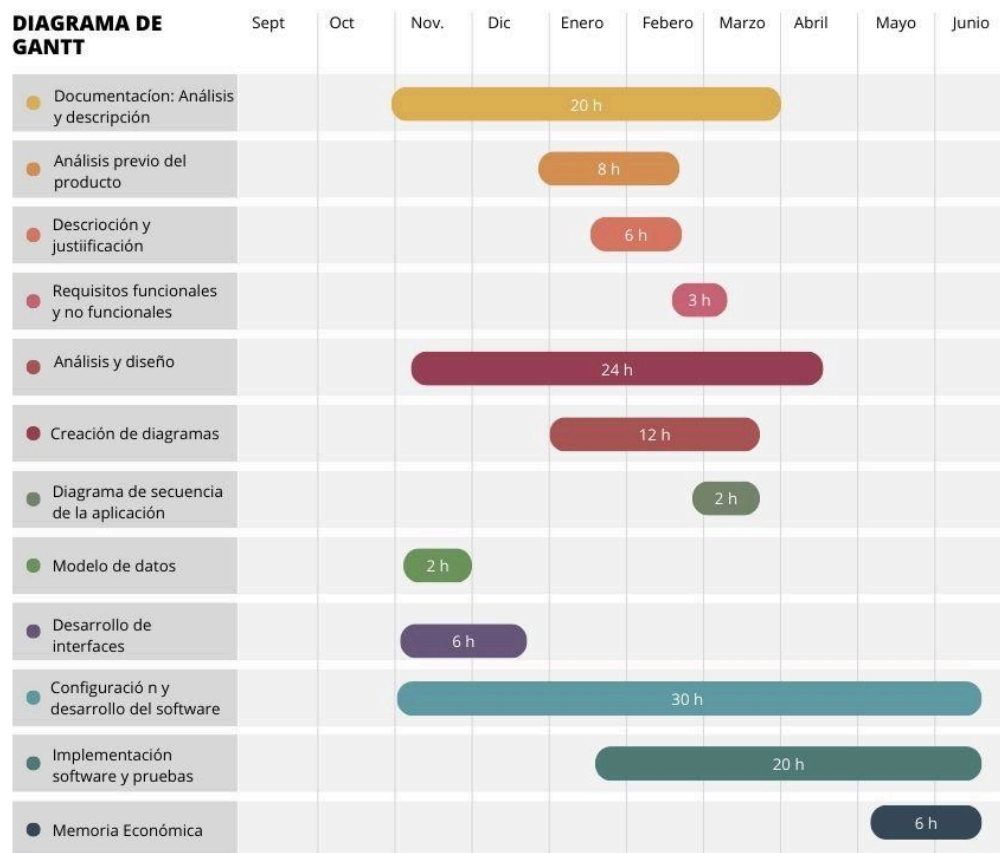


Figura 4. Diagrama de Gantt

En el Diagrama de Gantt se muestra el cronograma de los diferentes apartados desde septiembre hasta junio. Cada actividad está representada con diferentes colores y duraciones, indicando el tiempo estimado de su realización.

## 2. Análisis y diseño

### 2.1. Diagrama de Casos de Uso

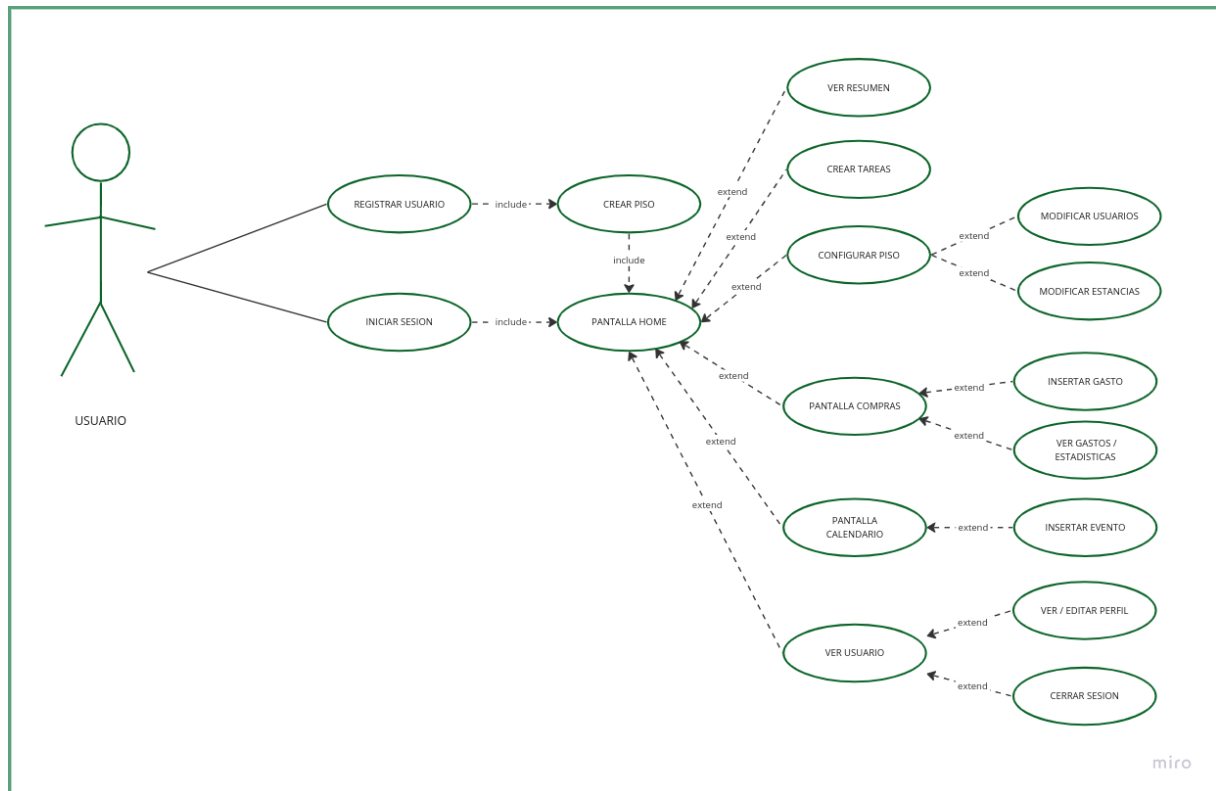


Figura 5. Diagrama de Casos de Uso

En el Diagrama de Casos de Uso se muestran las interacciones del usuario con el sistema. El usuario puede iniciar sesión y registrarse, lo cual incluye la creación de un Piso y acceder a la pantalla *home*. Desde la pantalla *home*, el usuario puede realizar diversas acciones extendidas como ver resumen, crear tareas, configurar piso, ver usuarios, y acceder a pantallas adicionales como compras y calendario.

## 2.2. Diagrama de Clases

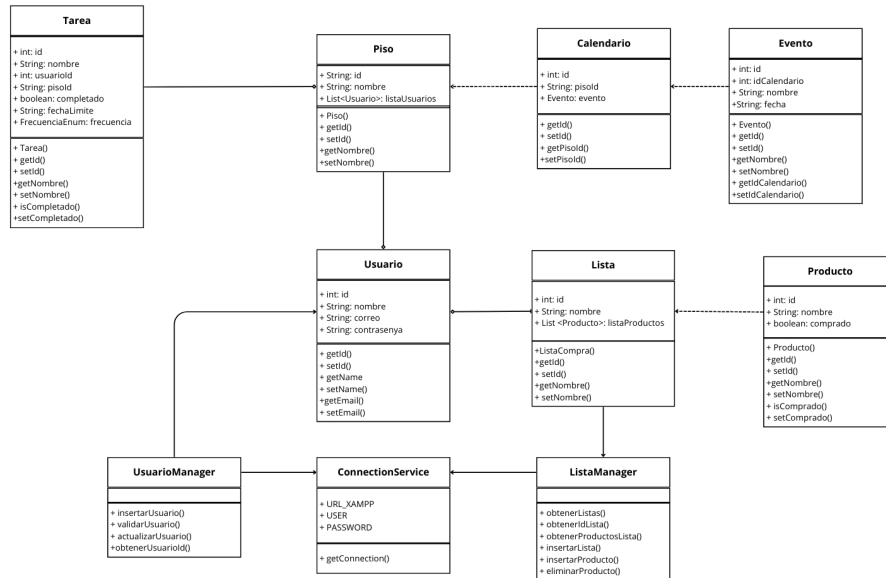
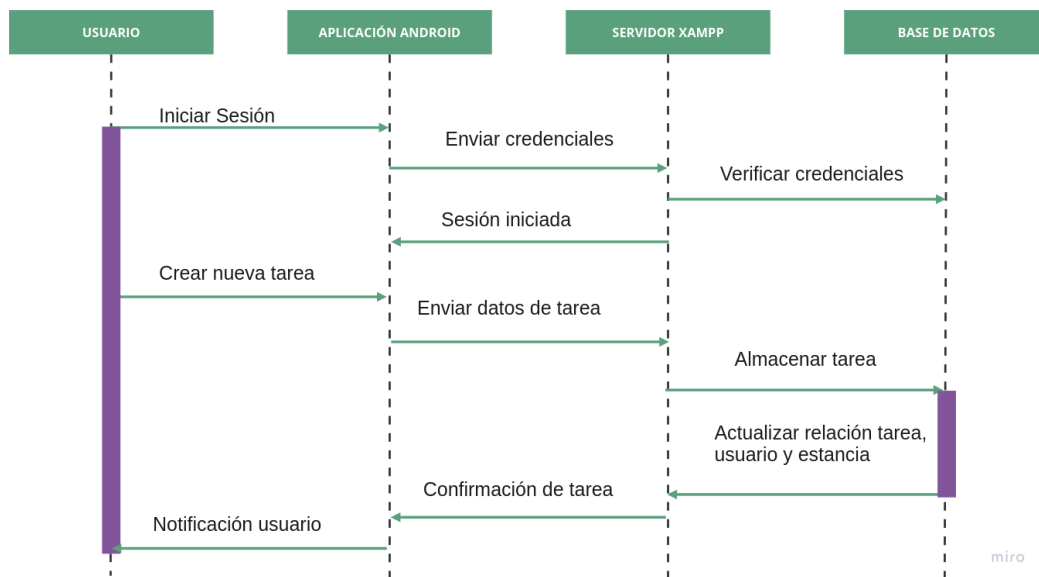


Figura.6 Diagrama de Clases

Este diagrama de clases representa una parte del proyecto, centrándose en las clases relacionadas con la gestión de tareas, pisos, calendarios, eventos, usuarios, listas de compras y productos. Se ha implementado una arquitectura más amplia con muchas más clases para abordar otros aspectos del proyecto.

## 2.3. Diagrama de Secuencia de la aplicación



*Figura 7. Diagrama de Secuencia de la Aplicación*

El Diagrama de Secuencia muestra el proceso de inicio de sesión y creación de una nueva tarea en la aplicación. El usuario envía sus credenciales a la aplicación Android, que las envía al servidor XAMPP para su verificación en la base de datos. Tras confirmar las credenciales, se inicia la sesión. El usuario crea una nueva tarea, cuyos datos se envían a través del servidor para ser almacenados en la base de datos. Finalmente, la relación de la tarea con el usuario y la estancia se actualiza, y el usuario recibe una confirmación de la tarea creada.

## 2.4. Modelos de datos

Realizar el esquema y el diagrama ha permitido planificar las relaciones entre las entidades y la estructura que tendrá la base de datos. También asegura que las relaciones entre tablas sean válidas y que las claves foráneas apunten a registros válidos.

### 2.4.1. Modelo lógico de datos

El modelo lógico de datos se ha diseñado para estructurar y organizar la información de manera clara y eficiente. A partir de este modelo, se ha podido elaborar tanto el diagrama entidad-relación (ER) como el diagrama relacional. Estos diagramas representan visualmente las entidades, atributos y relaciones, permitiendo una comprensión integral del sistema.

**PRODUCTOS** (#ID\_PRODUCTO, #ID\_LISTA, NOMBRE)

**LISTA\_COMPRA** (#ID\_LISTA, NOMBRE)

**USUARIO** (#ID\_USUARIO, #NOMBREUSUARIO, NOMBRE, APELLIDOS, EMAIL, CONTRASENYA)

**USUARIO\_LISTA** (#ID\_USUARIO, #ID\_LISTA, #ID\_PISO)

**COMPRA** (#ID\_COMPRA, #ID\_USUARIO, NOMBRE\_COMPRA, GASTO)

**TAREA** (#ID\_TAREA, #ID\_USUARIO, #ID\_ESTANCIA, NOMBRE\_TAREA, FECHA\_LIMITE: DATE, COMPLETADA: BOOLEAN)

**ESTANCIA** (#ID\_ESTANCIA, NOMBRE, #ID\_PISO)

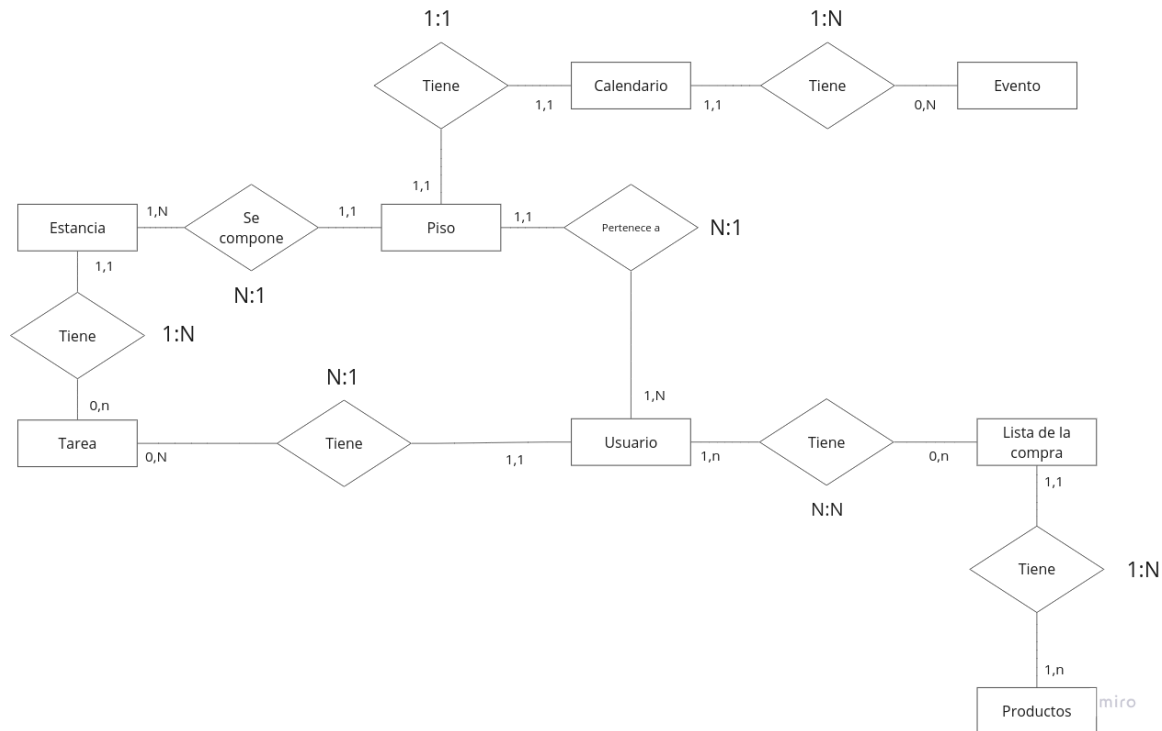
**PISO** (#ID\_PISO, NOMBRE)

**CALENDARIO** (#ID\_CALENDARIO, #ID\_PISO )

**EVENTO** (#ID\_EVENTO, #ID\_CALENDARIO, NOMBRE\_EVENTO, FECHA, HORA\_EMPIECE, HORA\_FIN)



#### 2.4.1.1. Diagrama Entidad-Relación

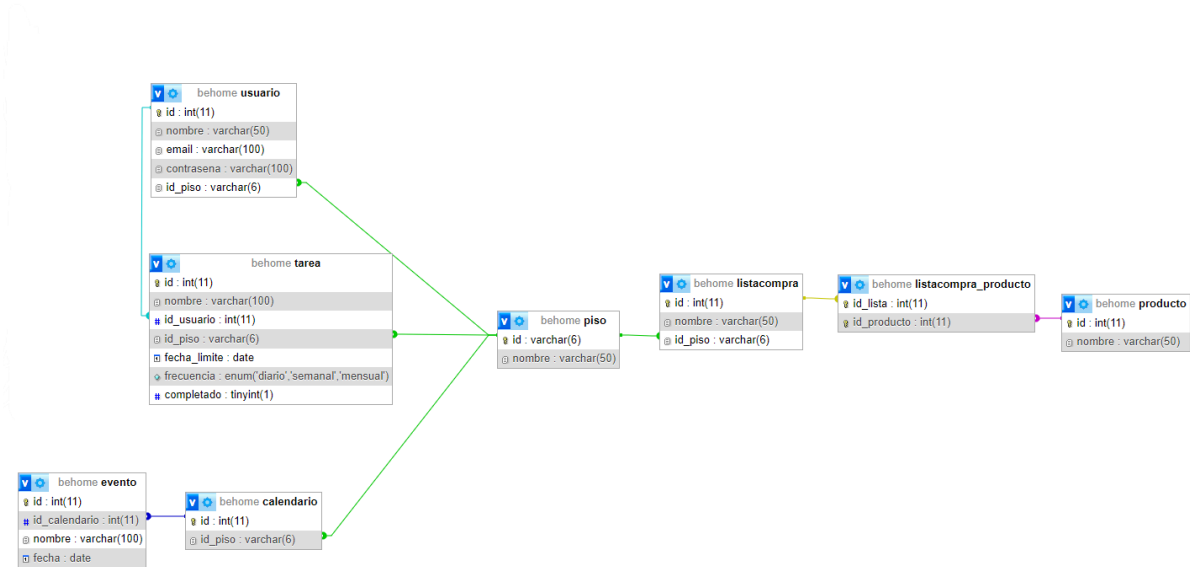


*Figura 8. Diagrama Entidad-Relación*

#### Relaciones:

- Un Producto puede pertenecer únicamente a una Lista
- Una Lista de la Compra puede tener ningún Producto o varios
- Una Lista de la Compra puede tener mínimo un Usuario o varios
- Un Usuario puede tener de 0 a varias Listas de la compra
- Un Usuario puede tener de 0 a varias Tareas asignadas.
- Una Tarea puede tener un único Usuario
- Una Tarea pertenece a una única Estancia
- Una Estancia tiene de 0 a varias Tareas asignadas
- Una Estancia pertenece a un único Piso
- Un Piso tiene de 1 a varias Estancias.
- Un Piso puede tener de 1 a varios Usuarios
- Un Usuario puede tener un único Piso
- Un Piso tiene un único Calendario
- Un Calendario pertenece a un único Piso
- Un Calendario puede tener de 0 a varios Eventos
- Un Evento pertenece a un único Calendario

#### 2.4.1.2. Diagrama Relacional

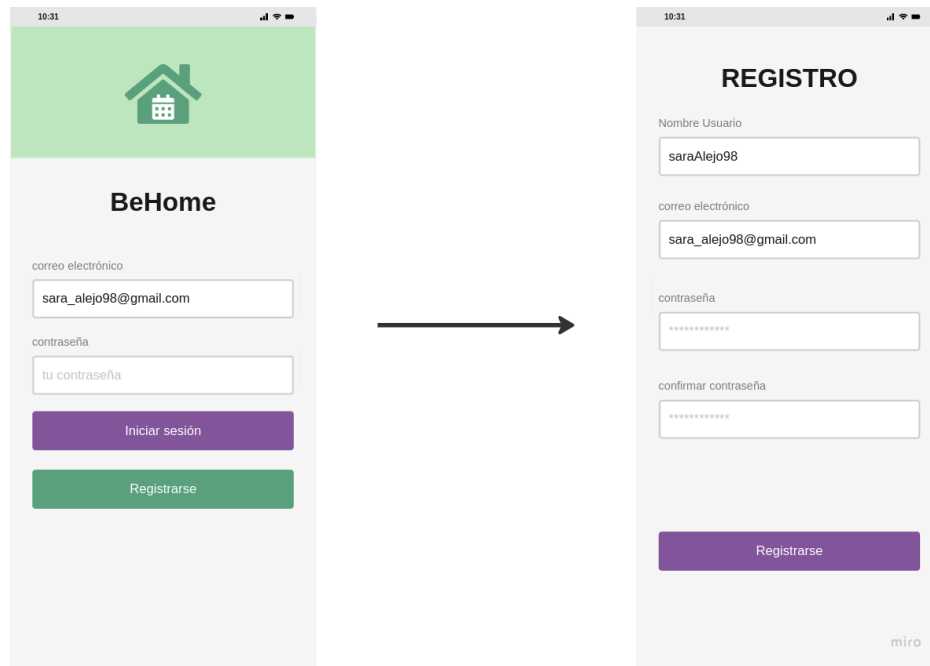


*Figura 9. Diagrama Relacional*

Aquí se pueden ver las relaciones con las claves primarias y foráneas de las tablas de la base de datos. También se pueden observar los cambios que se han realizado desde que se planteó la base de datos inicialmente, reflejando las modificaciones que surgieron durante el proceso de implementación y creación de la aplicación.

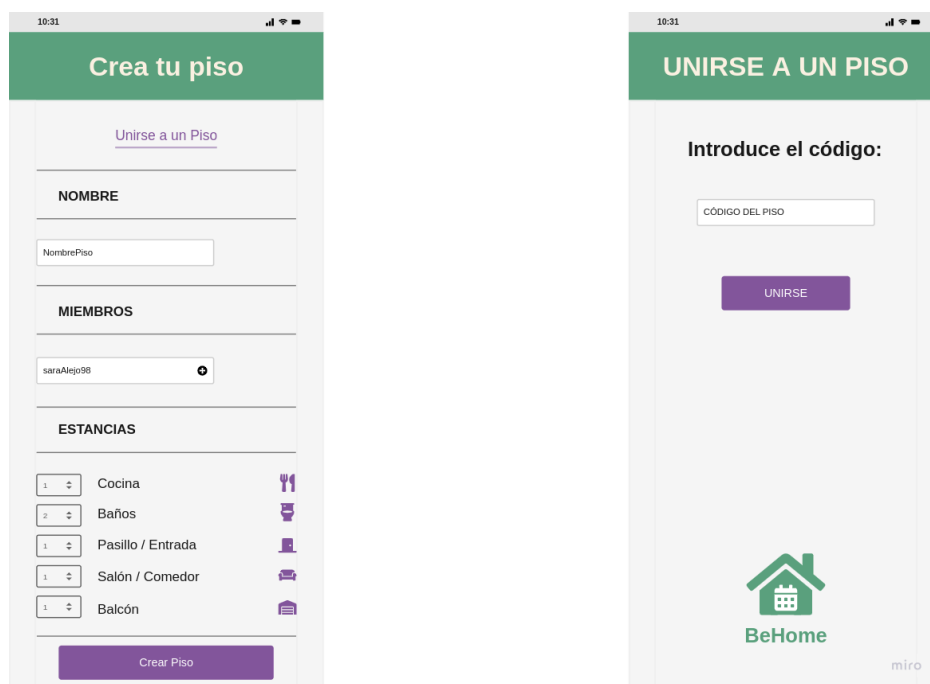
## 2.5. Desarrollo de Interfaces

Esta es la pantalla de inicio en la que se puede iniciar sesión o registrarse:



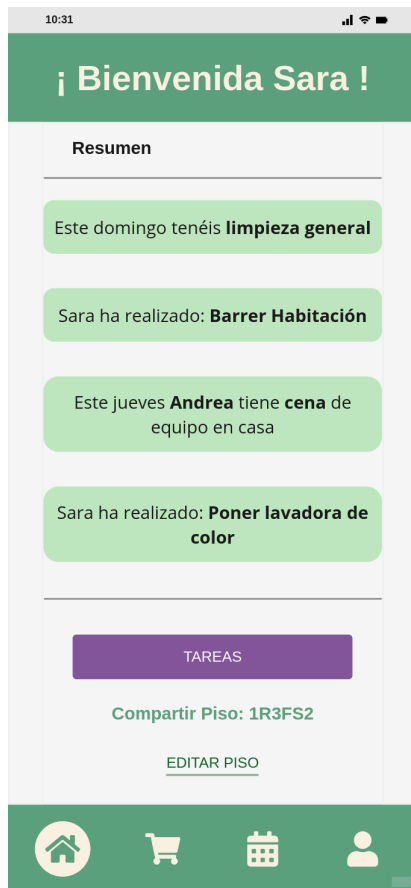
*Figura 10. Interfaz de Inicio de Sesión y Registro*

Una vez que el usuario se registra tiene dos opciones, crear un piso o unirse a otro:



*Figura 11. Interfaz de Crear un Piso y Unirse a un Piso*

Al iniciar sesión, la primera pantalla es la siguiente:



Una pantalla *HOME* en la que se muestra un pequeño resumen de la actividad del piso, como por ejemplo tareas o eventos.

Además también habrá un botón para poder ir a la pantalla de tareas.

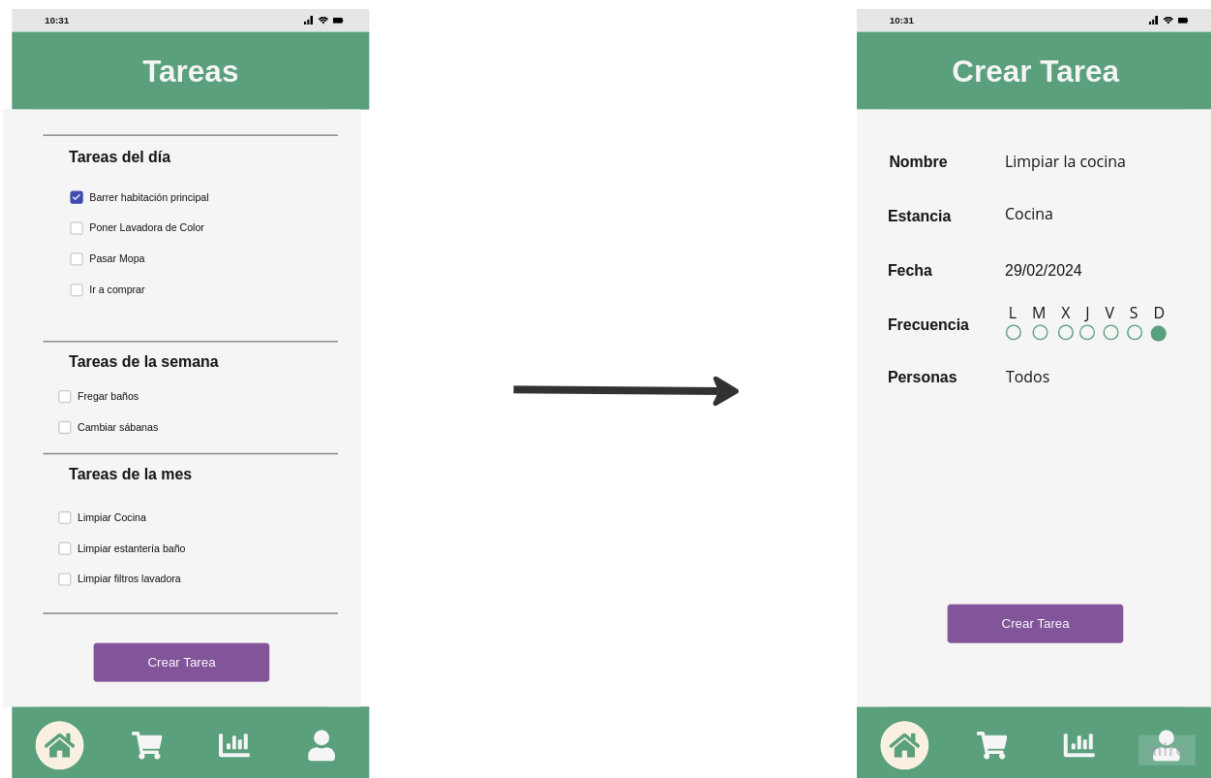
Se muestra el código del piso, para poder compartirlo y que otras personas puedan unirse.

También aparecerá un enlace para editar el piso en el que se puede cambiar el nombre o editar las estancias.

*Figura 12. Interfaz Principal de Resumen*

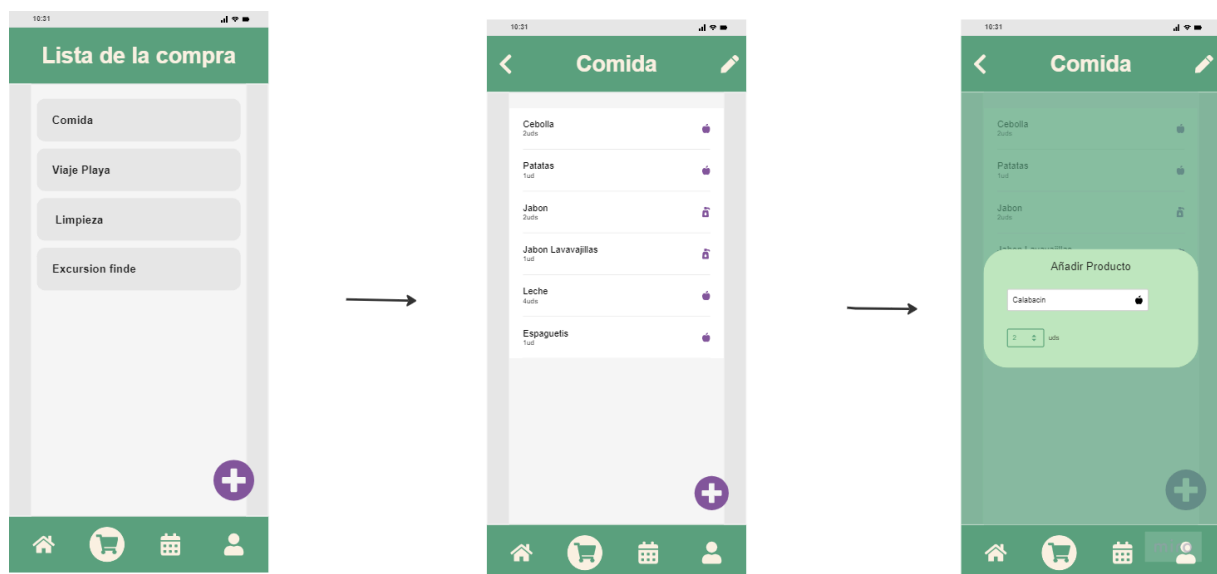
Si clicamos en el botón *TAREAS*, aparece una nueva pantalla en la que se pueden ver las diferentes tareas divididas por días, semana y mes.

Se podrán crear tareas ajustando el nombre, la estancia asociada, la fecha en la que hay que realizarla, la frecuencia con la que se quiere hacer y las personas asociadas.

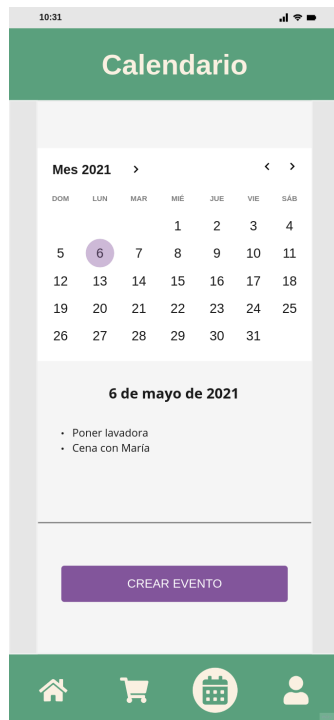


*Figura 13. Interfaz de Tareas y Crear Tarea*

Tenemos una barra de navegación inferior con las distintas opciones. Aquí se muestra la navegación del icono del carrito de la compra en la que se pueden gestionar las diferentes listas.



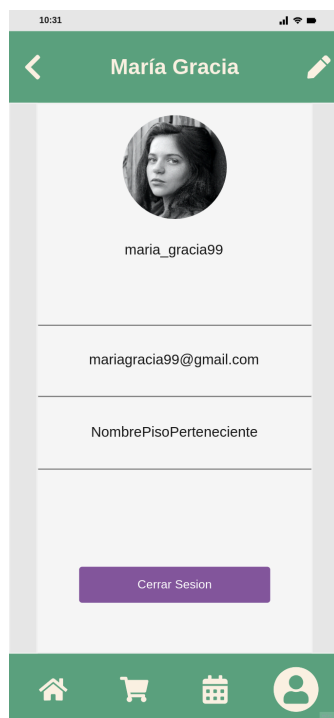
*Figura 14. Interfaz de Listas, Crear Lista y Añadir Producto*



También tenemos la pantalla del *CALENDARIO*:

Habrà un calendario mensual, un apartado con la fecha del día y las tareas o eventos que hay, además de un botón para crear eventos.

*Figura 15. Interfaz del Calendario*



Por último tenemos la pantalla del *PERFIL*:

Por ahora se ha planteado para que muestre los datos del usuario y pueda editarlos. También habrá un botón para cerrar sesión.

*Figura 16. Interfaz del Perfil*

## 3. Configuración y desarrollo del software

### 3.1. Implementación y pruebas

#### **Descripción de las clases extras añadidas durante la implementación**

Durante el desarrollo de la aplicación, se han identificado diferentes cambios y clases adicionales que no se habían contemplado en el diseño inicial. Estas clases son cruciales para la correcta implementación de la funcionalidad:

- Adaptadores de listas
  - Adaptadores para manejar la visualización de las tareas en una lista dentro de un RecyclerView. Este adaptador se encarga de enlazar los datos de las tareas con las vistas correspondientes.
- Fragments adicionales
  - Fragmentos para editar el piso, mostrar el resumen, la lista de tareas, gestionar y mostrar las listas con sus productos.

#### **Librerías utilizadas**

Se han empleado varias librerías y dependencias que han facilitado la implementación de las distintas funcionalidades:

- Navegación:
  - androidx.navigation:navigation-fragment:\$nav\_version
  - androidx.navigation:navigation-ui:\$nav\_version
- Gestión de la base de datos:
  - mysql-connector-java: 5.1.6 (Importante cambiar a esta versión específica para resolver problemas de conexión)

- Material Design:
  - androidx.compose.material3:material3: 1.2.1
  - androidx.compose.material3:material3-window-size-class: 1.2.1

## **Enlaces a tutoriales y libros**

Se han consultado diferentes tutoriales que han sido fundamentales para resolver problemas y aprender nuevas técnicas. A continuación, se listan algunos de los tutoriales más relevantes:

1. [Android Studio.-Base de datos con MySQL - CRUD COMPLETO](#)
2. [Save Data in Firebase Realtime Database in Android Studio || Firebase Realtime Database || 2021](#)
3. [CURSO COMPLETO ANDROID STUDIO 26: Fragments - explicación desde cero | Fragment vs Activity](#)
4. [How to Create Floating Action Button in Android Studio using Java | Android Knowledge](#)
5. [RecyclerView in Android Studio using Java | Android Knowledge](#)
6. [Custom ListView in Android Studio using Java](#)
7. [Weekly Calendar Android Studio Tutorial | Daily Events List](#)
8. [Number Picker Android Studio Tutorial - Populate Number Picker with Text](#)
9. [crear una base de datos en firebase](#)
10. [FIRESTORE Android 🔥 BASE de DATOS con Firebase](#)
11. [Cómo crear un LOGIN SCREEN en Android Studio ➡️📱](#)
12. [Diseño de Layouts - LinearLayout - Curso de Android en Kotlin](#)



13. [Curso Android desde cero #10 | Mi primer Aplicación en Android - Diseño lógico](#)

### **Comentarios sobre problemas importantes encontrados**

Conexión a la base de datos:

Después de muchas pruebas e intentos fallidos de arreglar la conexión con la base de datos, la solución fue cambiar la versión del conector a *mysql-connector-java 5.1.6*. Esto resolvió los problemas de compatibilidad y estabilidad en la conexión.

Gestión de hilos:

Implementar la gestión de hilos fue crucial ya que no se puede ejecutar todo en el hilo principal de la aplicación. Utilicé *AsyncTask* y posteriormente *Executors* para manejar operaciones en segundo plano, lo que mejoró el rendimiento y la capacidad de respuesta de la aplicación.

Navegación entre fragmentos:

Inicialmente, la navegación entre múltiples *Activities* causaba problemas de rendimiento. Decidí utilizar una única *Activity* principal con una barra de navegación inferior y múltiples *Fragments* para optimizar el rendimiento. Esto simplificó la navegación y mejoró la experiencia del usuario. Sin embargo, gestionar la comunicación y la navegación entre un gran número de *Fragments* resultó ser un desafío considerable.

Argumentos de Navegación (*Bundle*):

Se necesitaba pasar datos entre diferentes actividades y fragmentos, lo cual fue un reto al principio, ya que no paraban de aparecer errores.

## Diseño y Estructura de Layouts:

Organizar y diseñar los layouts ha sido todo un desafío, especialmente en fragmentos en los que había que organizar diferentes componentes.

## Obtener Identificadores de la Base de Datos con Generated Keys:

Al realizar inserciones en la base de datos, en varios modelos el identificador se creaba automáticamente, lo que al principio complicó la obtención y mapeo de este atributo. Se pudo solucionar con la implementación de *Generated Keys* que devuelve el identificador al ejecutar la query.

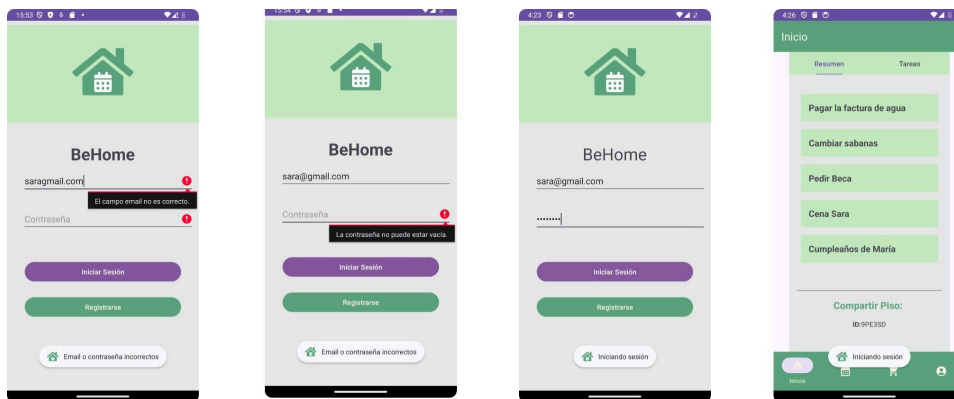
## **Funcionalidades que no han podido ser implementadas**

Debido a limitaciones de recursos, la implementación de Google Firebase y Firestore como base de datos no fue posible. Inicialmente se pudo desarrollar características clave como inicio de sesión, registro de usuarios, autenticación de Google y la capacidad de iniciar sesión mediante una cuenta de Google, además de implementar la creación de Pisos y almacenar información de usuarios en la base de datos.

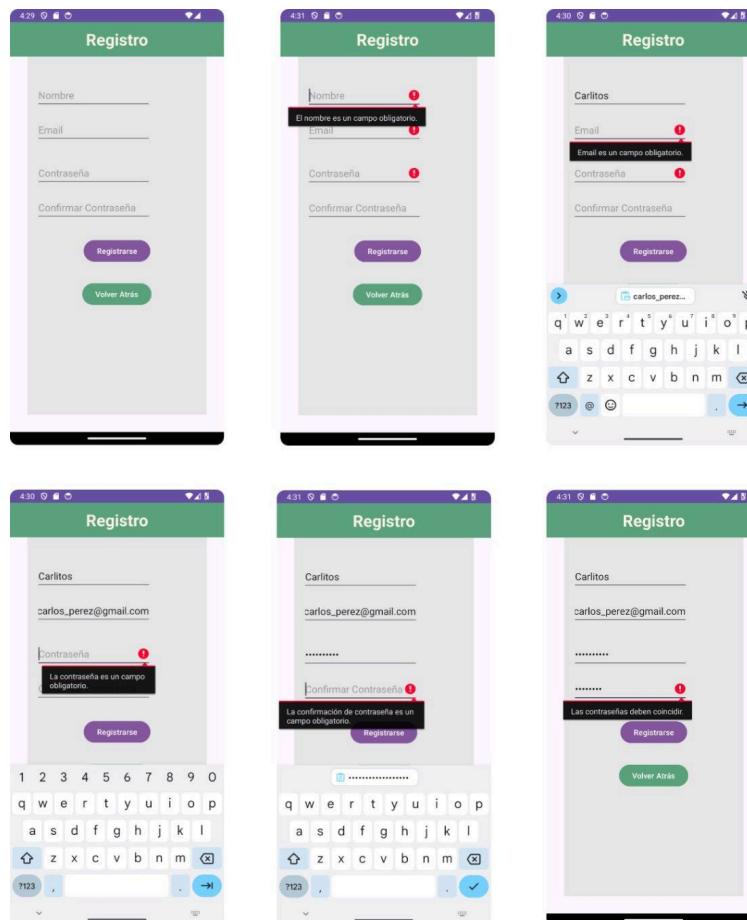
Sin embargo, a medida que el proyecto creció y se añadieron más clases, surgió una complejidad adicional debido a la falta de experiencia con bases de datos NoSQL, como Firestore. Esto complicó la gestión de la estructura de datos y las consultas, por lo que se decidió migrar hacia una base de datos SQL utilizando XAMPP, que facilitó el desarrollo del proyecto.

Además, se intentó que el usuario pudiera cargar una imagen para personalizar tanto el piso como su perfil. Sin embargo, debido a limitaciones técnicas, esta funcionalidad no pudo ser implementada.

## Capturas de pantalla para demostrar las pruebas realizadas:

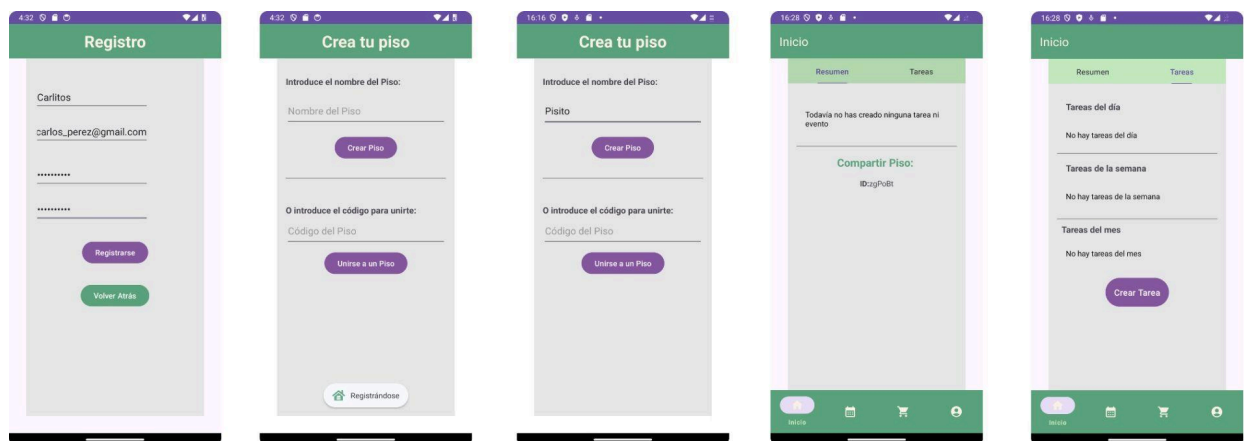


*Figura 17. Inicio de Sesión: Email incorrecto, Contraseña incorrecta, Credenciales correctas y Pantalla principal de la aplicación*

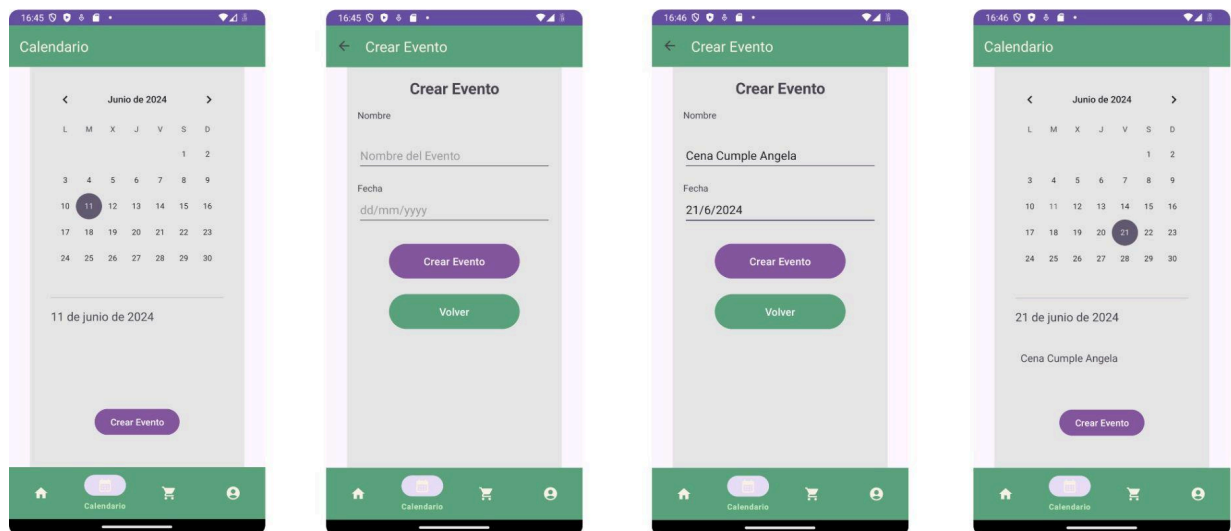


*Figura 18. Pantallas de registro incorrecto*

*BeHome: Desarrollo de una Aplicación Móvil para la Gestión del Hogar*  
*Sara Alejo Millán*

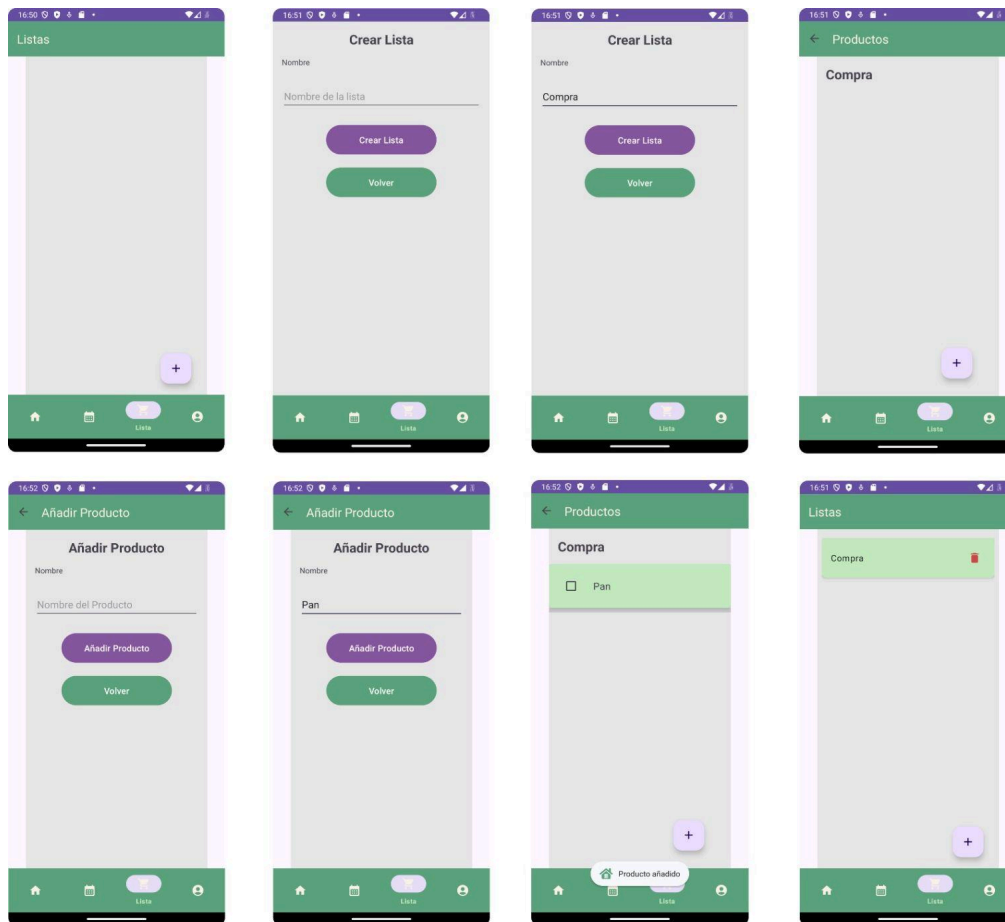


*Figura 19. Registro correcto y creación de un piso*

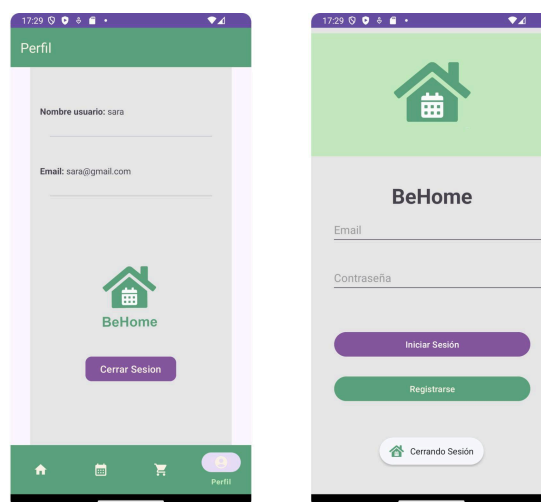


*Figura 20. Vista del Calendario y creación de un Evento*

*BeHome: Desarrollo de una Aplicación Móvil para la Gestión del Hogar*  
*Sara Alejo Millán*



*Figura 21. Creación de una lista y sus productos*



*Figura 22. Vista del Perfil y Cierre de Sesión*

## **Funcionalidades a añadir en el futuro**

Algunas funcionalidades no han podido ser implementadas debido a limitaciones técnicas y temporales. Sin embargo, podrían ser añadidas en el futuro:

1. **Acceso con Google y huella dactilar:** Integrar métodos de autenticación adicionales para mejorar la seguridad y la conveniencia del usuario.
2. **Foto de perfil y de Piso:** Permitir a los usuarios añadir fotos de perfil y fotos de los pisos, personalizando así su experiencia.
3. **Base de datos de productos para la lista de la compra:** Crear una base de datos para los productos que permita a los usuarios seleccionar productos predeterminados o guardar productos personalizados para añadirlos fácilmente a sus listas de compras.
4. **Contactar con empresas de limpieza o supermercados:** Incluir un apartado para contactar directamente con empresas de limpieza o supermercados locales y realizar pedidos desde la app.
5. **Mejoras en la frecuencia y asignación de tareas:** Optimizar la lógica para la frecuencia y asignación de tareas, mejorando la funcionalidad y usabilidad de la aplicación.
6. **Refactorización del código:** Planificar un refactor del código para mejorar su mantenibilidad y legibilidad, dado que la prioridad inicial fue la funcionalidad.
7. **Mejora del diseño:** Revisar y mejorar las interfaces para que sean más atractivas visualmente y accesibles, siguiendo principios de diseño modernos y accesibles.
8. **Notificaciones en tiempo real:** Implementar notificaciones push para alertar a los usuarios sobre nuevas tareas o cambios en las tareas existentes.
9. **Soporte multi-idioma:** Extender el soporte de la aplicación a múltiples idiomas para llegar a una audiencia más amplia.

## 3.2. Memoria económica

### Recursos Materiales

#### 1. Hardware:

- a. Ordenadores de desarrollo: Se utilizó 1 ordenador portátil con un coste aproximado de 800 €.
- b. Monitores externos: Se utilizaron 2 monitores con un coste unitario de 150 €
- c. Periféricos como el teclado, ratón, cables etc.

#### 2. Software

- a. IDE de desarrollo: Android Studio 2022.3
- b. Conector MySQL: MySQL Connector
- c. XAMPP para gestionar la base de datos
- d. Librerías y dependencias: Utilización de librerías open-source.

### Recursos Humanos

El desarrollo de la aplicación BeHome fue realizado por un equipo compuesto por un desarrollador principal durante el periodo de marzo a junio.

Horas trabajadas: 600 horas para el desarrollo de la aplicación. Coste por hora: 25 €/hora.

### Coste Fijos del Proyecto el primer año

Costes Fijos	Coste Mensual	1er Año
Alquiler Local	300 €	3.600 €
Recursos Materiales	-	3.036 €
Seguro	75 €	900 €
Suministros Básicos	180 €	2.160 €
Marketing y publicidad	70 €	840 €
Gestoría	50 €	600 €
Nomina	1.400 €	16.800 €
Seguridad Social	490 €	5.880 €
Cuota Autónomos	80 €	960 €
<b>TOTAL</b>		<b>34.776 €</b>

*Figura 23. Tabla de Costes Fijos*

Aquí se muestran los Recursos Materiales desglosados:

Recursos Materiales	Precio
<b>Hardware</b>	
Ordenador portátil	800 €
Monitores externos	300 €
Periféricos (teclados, ratón, cables...)	100 €
<b>Software</b>	
Google Play Store	25 €
AppStore	91 €
Servidores	100 €
<b>Otros</b>	
Escritorios	200 €
Silla	120 €
<b>TOTAL</b>	<b>1.736 €</b>

*Figura 24. Tabla de Recursos Materiales*



### **Análisis de Rentabilidad**

Conociendo el coste total del proyecto, es posible analizar la viabilidad económica del proyecto y su rentabilidad. Para ello, se pueden considerar diversos planes de negocio, como la monetización mediante suscripciones, publicidad o venta de funcionalidades premium. A continuación, se presentan algunas estrategias:

- **Suscripciones:** Ofrecer una suscripción mensual para acceder a funcionalidades avanzadas. Si se estima una base de 500 usuarios con una suscripción mensual de 1,99 €, se obtendrían 995 € al mes.
- **Publicidad:** Incluir publicidad en la aplicación puede generar ingresos adicionales. Suponiendo unos ingresos medios de 0.10 € por usuario al día y una base de 1.000 usuarios, se podrían obtener 3.000 € mensuales.
- **Funcionalidades Premium:** Vender funcionalidades avanzadas o adicionales por un precio fijo.


## 4. Conclusión

Este proyecto ha sido desarrollado con el objetivo de simplificar la gestión del hogar compartido mediante una aplicación móvil intuitiva y eficiente. Esta aplicación se enfoca en mejorar la convivencia al proporcionar herramientas prácticas para la organización de tareas, gestión de gastos y comunicación efectiva entre los miembros del hogar. BeHome facilita la asignación y seguimiento de tareas domésticas, la planificación de compras mediante listas compartidas y la coordinación de eventos a través de un calendario común. La implementación de estas funcionalidades está orientada a fomentar la colaboración, transparencia y una mejor organización diaria, creando un ambiente más armónico y eficiente en los hogares compartidos.


Durante el desarrollo de la aplicación, se han enfrentado y resuelto diversos desafíos técnicos, tales como la integración de bases de datos, gestión de hilos y optimización de la navegación entre interfaces. La aplicación ha sido diseñada con un enfoque en la usabilidad y accesibilidad, utilizando tecnologías como Java, Android Studio y bases de datos SQL. Además, se han considerado aspectos importantes como el rendimiento para asegurar una experiencia de usuario óptima. A través de un análisis de mercado y un enfoque práctico, podríamos afirmar que BeHome se posiciona como una solución competitiva y efectiva en el mercado de aplicaciones de gestión del hogar, proporcionando una herramienta que mejora significativamente la vida en comunidad para jóvenes y adultos que comparten vivienda.

## 5. Bibliografía

Android Knowledge, (2022a). RecyclerView in Android Studio using Java [en línea]. *Youtube*. [Consultado el 4 de junio de 2024]. Disponible en:

 [RecyclerView in Android Studio using Java | Android Knowledge](#)

Android Knowledge, (2022b). How to Create Floating Action Button in Android Studio using Java [en línea]. *YouTube*. [Consultado el 4 de junio de 2024]. Disponible en:

 [How to Create Floating Action Button in Android Studio using Java | Android Knowledge](#)

Android Developers, (2023a). Bottom Navigation View [en línea]. *Android Developers*. [Consultado el 4 de junio de 2024]. Disponible en:

[BottomNavigationView | Android Developers](#)

Android Developers, (2023b). Material CheckBox [en línea]. *Android Developers*. [Consultado el 4 de junio de 2024]. Disponible en:

[MaterialCheckBox | Android Developers](#)

Android Developers, (2024). Number Picker [en línea]. *Android Developers*. [Consultado el 4 de junio de 2024]. Disponible en:

[NumberPicker | Android Developers](#)

Google Developers, (2023). Cómo usar SQL para leer y escribir en una base de datos [en línea]. *Android Developers*. [Consultado el 4 de junio de 2024]. Disponible en: [Cómo usar SQL para leer y escribir en una base de datos](#)

Google Developers, (2024a). Firebase Authentication [en línea]. *Firebase*. [Consultado el 4 de junio de 2024]. Disponible en: <https://firebase.google.com/docs/auth?hl=es>

Google Developers, (2024b). Firebase Realtime Database [en línea]. *Firebase*. [Consultado el 4 de junio de 2024]. Disponible en: <https://firebase.google.com/docs/database?hl=es>

Google Developers, (2024c). Firestore [en línea]. *Firebase*. [Consultado el 4 de junio de 2024]. Disponible en: <https://firebase.google.com/docs/firestore?hl=es>

Oracle Java Documentation, (sin fecha). The Java Tutorials: Lambda Expressions [en línea]. *Oracle*. [Consultado el 5 de junio de 2024]. Disponible en: <https://docs.oracle.com/javase/tutorial/java/javaOO/lambdaexpressions.html>

Programación Android by AristiDevs, (2020). Diseño de Layouts - LinearLayout - Curso de Android en Kotlin [en línea]. *YouTube*. [Consultado el 4 de junio de 2024]. Disponible en: [▶ Diseño de Layouts - LinearLayout - Curso de Android en Kotlin](#)