
# UNIVERSIDAD PRIVADA DE TACNA

## FACULTAD DE INGENIERÍA

## Escuela Profesional de Ingeniería de Sistemas

# *"App de Gestión Financiera para el Registro y Análisis de Gastos Personales"*

Curso: Patrones de Software

Docente: Ing. Patrick Cuadros

Integrantes:

***Ayma Choque, Erick Yoel (2021072616)***
***Poma Machicado, Fabiola Estefani (2021070030)***
***Tapia Vargas, Dylan Yariet (2021072630)***

**Tacna – Perú**
**2025**

| CONTROL DE VERSIONES | | | | | |
|---|---|---|---|---|---|
| Versión | Hecha por | Revisada por | Aprobada por | Fecha | Motivo |
| 1.0 | MPV | ELV | ARV | 11/06/2025 | Versión Original |

# App de Gestión Financiera para el Registro y Análisis de Gastos Personales
# Documento de Arquitectura de Software

**Versión *{1.0}***

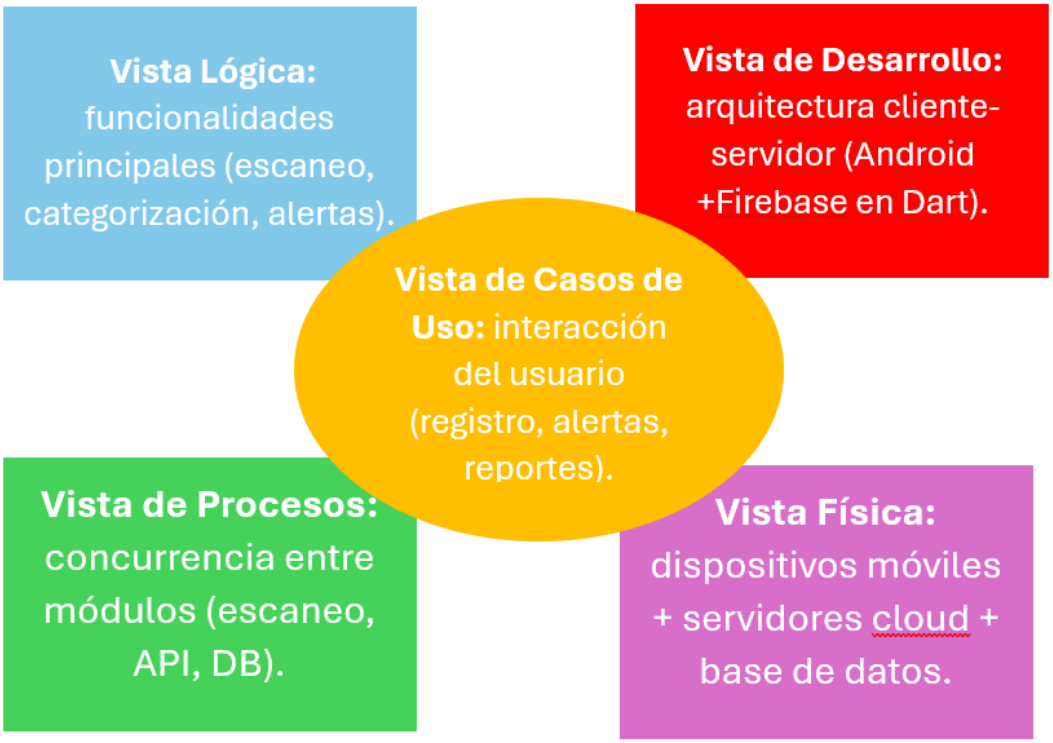
Digix

| CONTROL DE VERSIONES | | | | | |
|---|---|---|---|---|---|
| Versión | Hecha por | Revisada por | Aprobada por | Fecha | Motivo |
| 1.0 | MPV | ELV | ARV | 10/10/2020 | Versión Original |

## Contenido

# 1. INTRODUCCIÓN

## 1.1. Propósito (Diagrama 4+1)




## 1.2. Alcance

Este proyecto tiene como objetivo desarrollar y mejorar una aplicación móvil de gestión financiera personal llamada VanguardMoney, que permita a los usuarios administrar sus finanzas de forma segura y eficiente mediante las siguientes funcionalidades clave:

Gestión de Finanzas Personales:

- ➢ Registro automático de egresos mediante análisis de facturas con IA (Google Gemini AI)
- ➢ Escaneo inteligente de facturas por imagen y análisis de texto
- ➢ Categorización automática de gastos (Alimentos, Hogar, Ropa, Salud, Tecnología, Entretenimiento, Transporte, Mascotas, Otros)
- ➢ Planificación presupuestaria con límites por categoría y seguimiento en tiempo real
- ➢ Control de gastos diarios con configuración de límites personalizados
- ➢ Gestión de ahorros integrada con planes financieros

Digix

Análisis y Reportes:

- Dashboard de inicio con resumen financiero en tiempo real
- Análisis avanzado con gráficos de barras, líneas y circulares
- Reportes por períodos (semanal, mensual, anual)
- Filtros personalizables por fechas, categorías y rangos
- Visualización de tendencias de gastos y patrones de consumo

Seguridad y Autenticación:

- Autenticación Firebase con correo electrónico y contraseña
- Recuperación de contraseña por correo electrónico
- Gestión segura de datos en Firebase Firestore
- Validación de formularios con controles personalizados

Tecnologías Implementadas:

- Flutter como framework principal
- Firebase (Authentication, Firestore, Storage)
- Google Gemini AI para análisis de facturas
- Provider para manejo de estado
- FL Chart para visualizaciones
- Image Picker para captura de imágenes

Escalabilidad y Rendimiento:

- Arquitectura modular con separación de responsabilidades (presentación, lógica de negocio, datos)
- Optimización de consultas Firebase para manejar crecimiento de usuarios
- Manejo eficiente de imágenes y procesamiento IA
- Estructura preparada para futuras expansiones funcionales

Este alcance refleja una aplicación enfocada en la gestión personal automatizada de gastos con inteligencia artificial, diferenciándose por su capacidad de análisis automático de facturas y planificación presupuestaria inteligente.

## 1.3. Definición, siglas y abreviaturas

Definición de Términos Clave

- Usuario: Persona que interactúa con la plataforma digital para gestionar sus finanzas personales.
- Plataforma Digital: Conjunto de aplicaciones móviles y web diseñadas para la gestión de finanzas personales.
- Transacción: Operación financiera registrada por el usuario, ya sea de ingreso o egreso.
- Presupuesto: Límite establecido por el usuario para controlar sus gastos en diferentes categorías.
- Reportes Financieros: Informes generados por la plataforma que muestran un resumen de las finanzas del usuario, como ingresos, egresos y balance general.
- Notificaciones Push: Alertas enviadas a los usuarios en tiempo real sobre eventos relevantes, como gastos excesivos o pagos recurrentes.

Siglas y Abreviaturas

- RF: Requerimiento Funcional.
- RNF: Requerimiento No Funcional.
- MFA: Autenticación Multi Factor (Multi-Factor Authentication).
- CSV: Comma Separated Values (Valores Separados por Comas).
- PDF: Portable Document Format (Formato de Documento Portátil).
- API: Interfaz de Programación de Aplicaciones (Application Programming Interface).
- UX/UI: Experiencia de Usuario / Interfaz de Usuario (User Experience / User Interface).

Digix

# 2. OBJETIVOS Y RESTRICCIONES ARQUITECTÓNICAS

## 2.1. Priorización de requerimientos

### 1.1.1. Requerimientos Funcionales

| RF | Requerimiento | Descripción |
|---|---|---|
| RF01 | Iniciar Sesión | El usuario ingresa email y contraseña para autenticarse. |
| RF02 | Registrar Usuario | El usuario crea una cuenta con email, contraseña, nombre, teléfono y foto de perfil. |
| RF03 | Recuperar Contraseña | El usuario solicita recuperar su contraseña. Luego recibe un correo para cambiarla. |
| RF04 | Editar Perfil | El usuario modifica su información personal y foto de perfil. |
| RF05 | Configurar Gasto Diario | El usuario establece un límite diario de gasto con alertas si se excede. |
| RF06 | Editar Gasto Diario | El usuario modifica el límite diario de gasto |
| RF07 | Crear Planes de Gastos (Mensuales) | El usuario establece presupuestos mensuales por categoría. |
| RF08 | Editar Planes | El usuario modifica presupuestos existentes. |
| RF09 | Alertar Sobre Gasto Excesivo | La app notifica si se sobrepasa un presupuesto. |
| RF10 | Registrar gasto manualmente | El usuario ingresa los datos del gasto manualmente. Luego guarda el registro. |
| RF11 | Registrar gasto por escaneo | El usuario toma o carga una foto del recibo. La app usa IA para extraer y categorizar los datos. El usuario confirma antes de guardar. |
| RF12 | Visualizar Análisis Financiero | El usuario ve gráficos y estadísticas de sus gastos. |
| RF13 | Crear Deudas y Préstamos | El usuario registra deudas pendientes y pagos programados. |
| RF14 | Editar Deudas y Préstamos | El usuario modifica deudas pendientes y pagos programados. |
| RF15 | Notificar Pagos y Vencimientos | La app envía recordatorios sobre pagos próximos. |
| RF16 | Exportar Reportes y Movimientos | El usuario puede descargar reportes y sus movimientos financieros en formatos Excel y PDF. |

### 1.1.2. Requerimientos No Funcionales – Atributos de Calidad

| RNF | Requerimiento | Descripción | Prioridad |
|---|---|---|---|
| RNF01 | Rendimiento | El aplicativo debe responder rápidamente en todas las operaciones comunes (registrar transacciones, generar reportes, mostrar gráficos) sin demoras notables, con tiempos de carga no mayores a 2 segundos. | Alta |
| RNF02 | Seguridad | Los datos del usuario (como contraseñas y transacciones) deben estar protegidos mediante encriptación robusta (como AES-256) y protocolos de seguridad actualizados (como HTTPS y autenticación multifactor). | Alta |
| RNF03 | Usabilidad | La interfaz del aplicativo debe ser intuitiva y fácil de usar, permitiendo a los usuarios completar tareas como ingresar transacciones y visualizar reportes sin complicaciones. La navegación debe ser fluida tanto en dispositivos móviles como en escritorio. | Alta |
| RNF04 | Escalabilidad | El aplicativo debe ser capaz de manejar un número creciente de usuarios y transacciones sin afectar su rendimiento. Se debe poder escalar en la base de datos y en el manejo de reportes sin comprometer la experiencia del usuario. | Media |
| RNF05 | Disponibilidad | El sistema debe estar disponible la mayor parte del tiempo, con un mínimo de 99.5% de tiempo de funcionamiento sin interrupciones significativas. | Alta |
| RNF06 | Mantenimiento | El aplicativo debe permitir actualizaciones y mejoras continuas sin que los usuarios experimenten interrupciones significativas. La implementación de nuevas funcionalidades no debe requerir reescritura del código base. | Media |
| RNF07 | Notificaciones | El aplicativo debe permitir la configuración de notificaciones push para alertas personalizadas (gastos excesivos, presupuesto, pagos recurrentes) que se envíen en tiempo real, asegurando que el usuario esté siempre al tanto de sus finanzas. | Baja |

2.2. Restricciones

- **Compatibilidad limitada**:
  Solo funcionará en dispositivos móviles con Android 10 o superior.

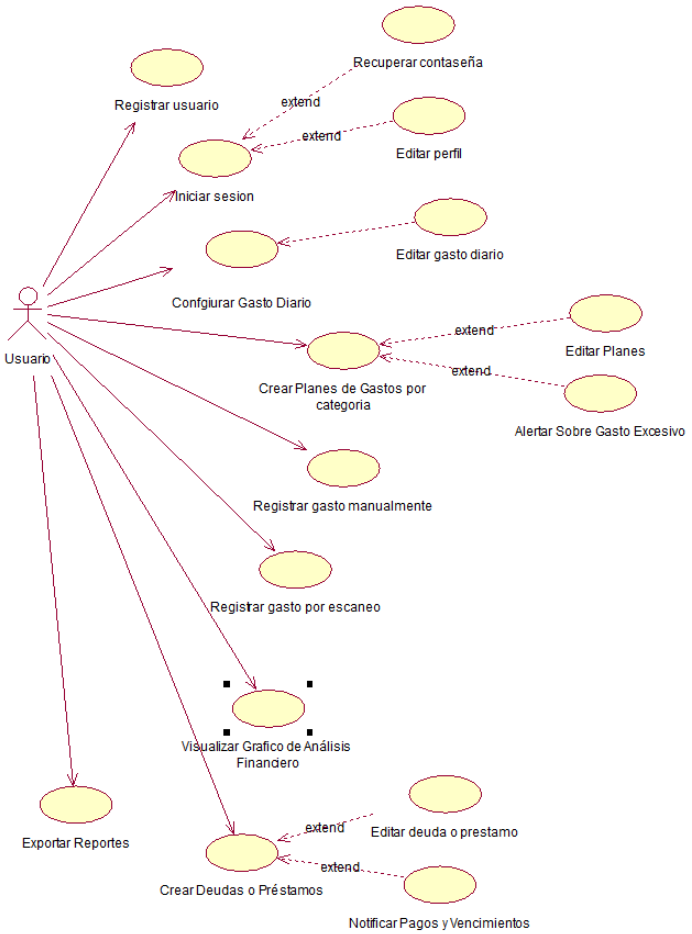
- **Conectividad obligatoria**:
  Requiere conexión a Internet para sincronización y procesamiento de imágenes (a través de la API Gemini).

- **Calidad de imagen crítica**:
  La precisión del reconocimiento automático depende de la nitidez y formato del documento escaneado.

- **Cumplimiento normativo**:
  Debe ajustarse a las políticas de Google Play, GDPR, CCPA y otras normativas legales de privacidad de datos.

# 3. REPRESENTACIÓN DE LA ARQUITECTURA DEL SISTEMA

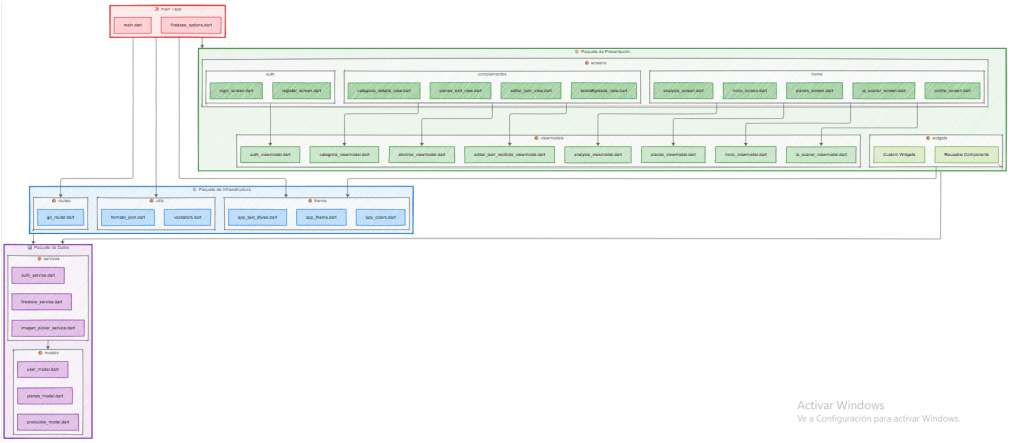
### 1.1.3. Diagramas de Casos de uso



## 3.1. Vista Lógica

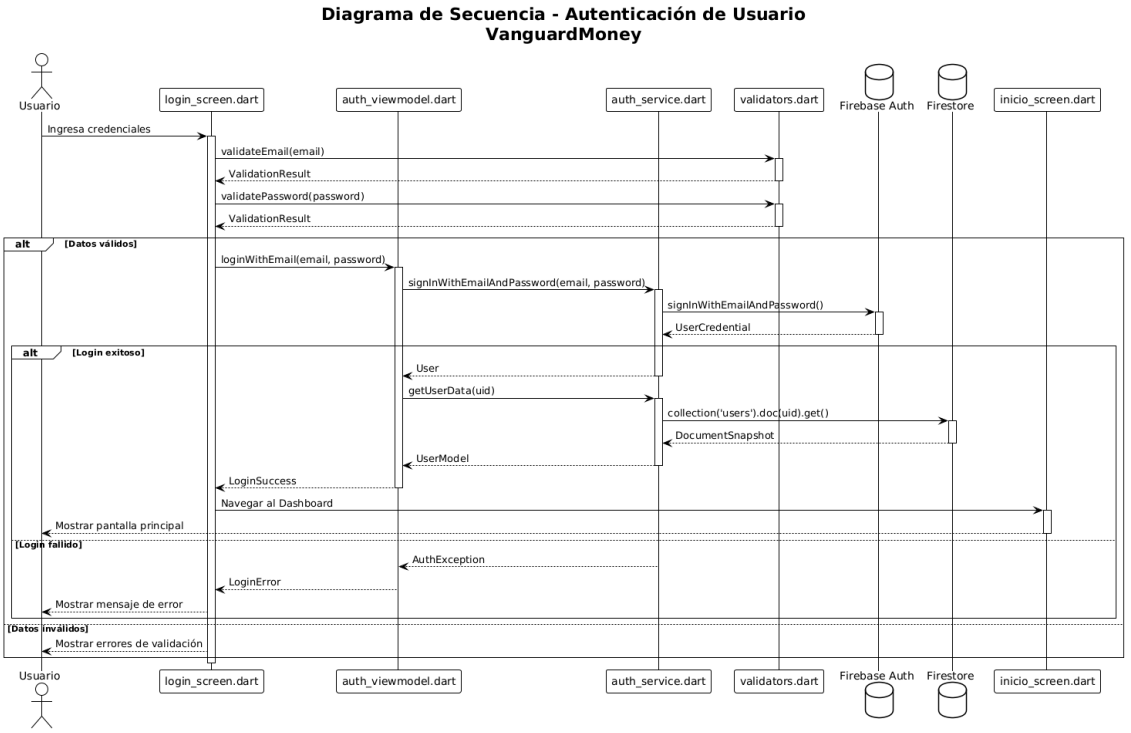
### 3.1.1. Diagrama de Subsistemas (paquetes)

https://www.mermaidchart.com/app/projects/4fa7b273-3e12-4cd1-a6e2-5ce82b1b910d/diagrams/5144ca63-964b-4864-85ec-5342ee0fcedd/version/v0.1/edit

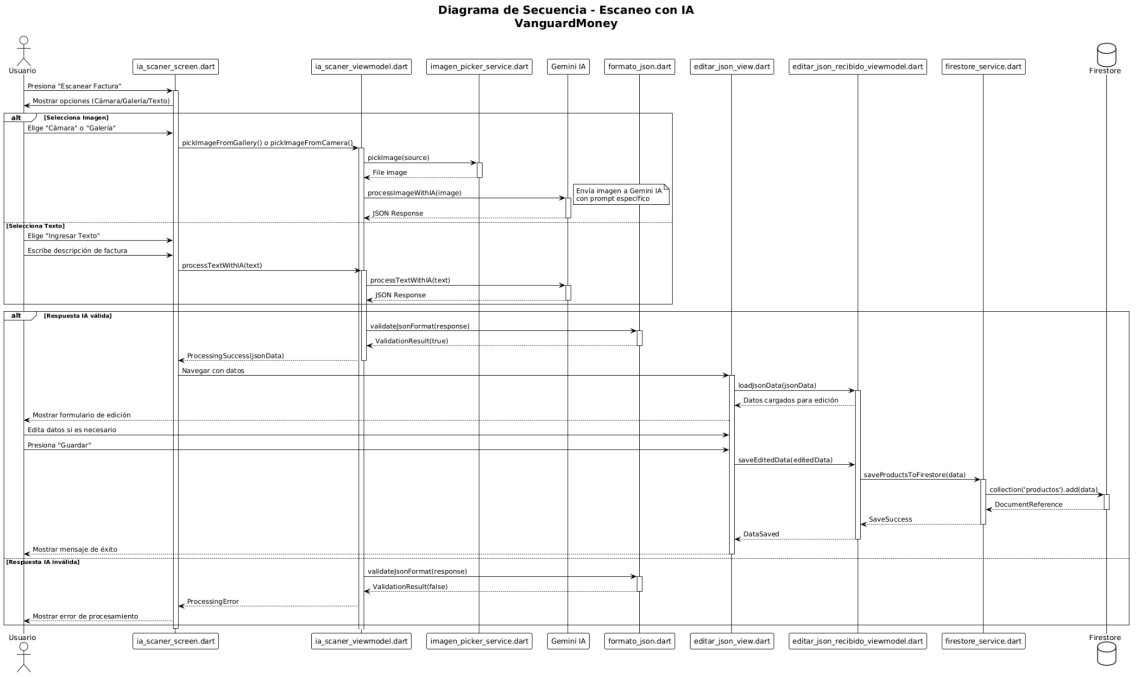


### 3.1.2. Diagrama de Secuencia (vista de diseño)

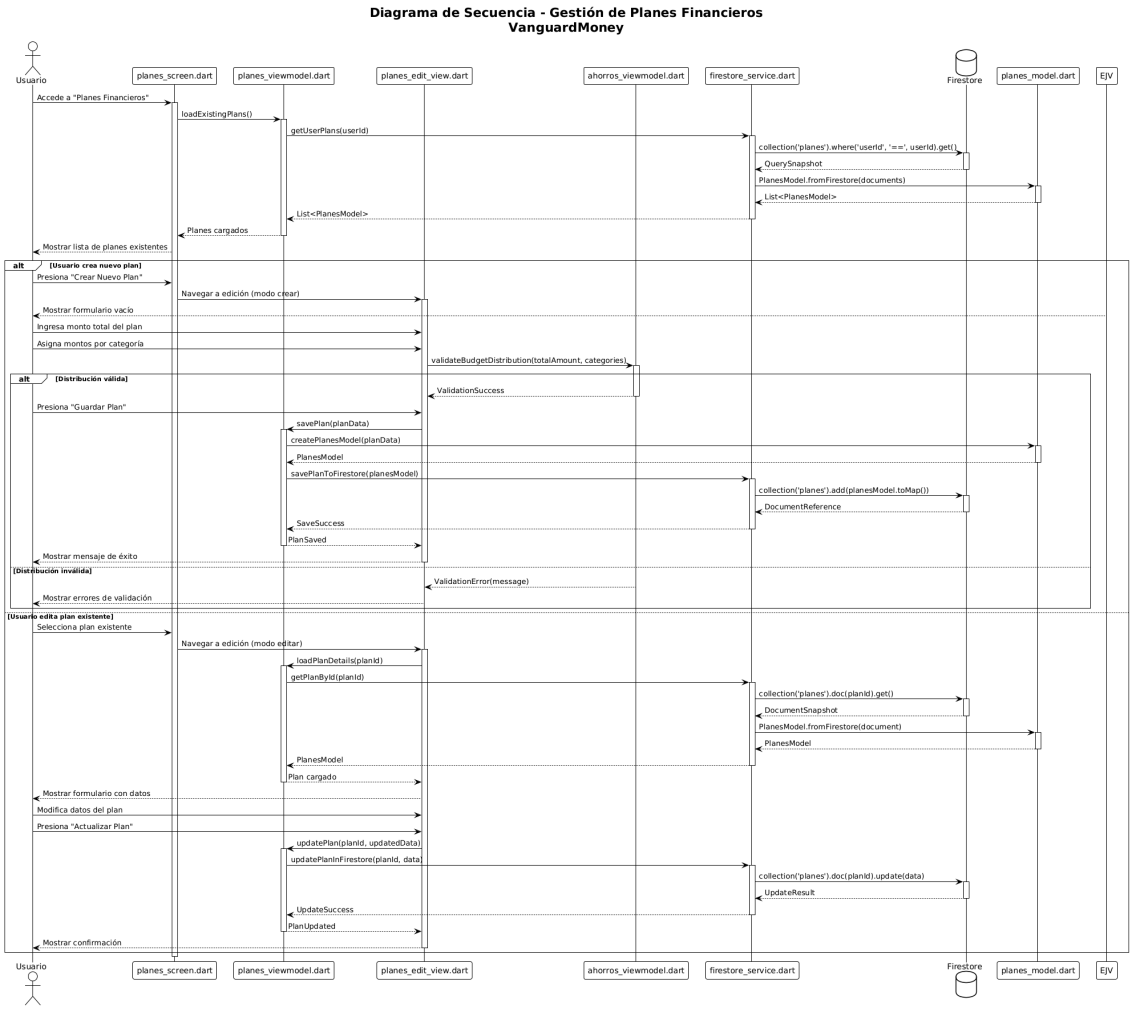
1. Secuencia de Autenticación



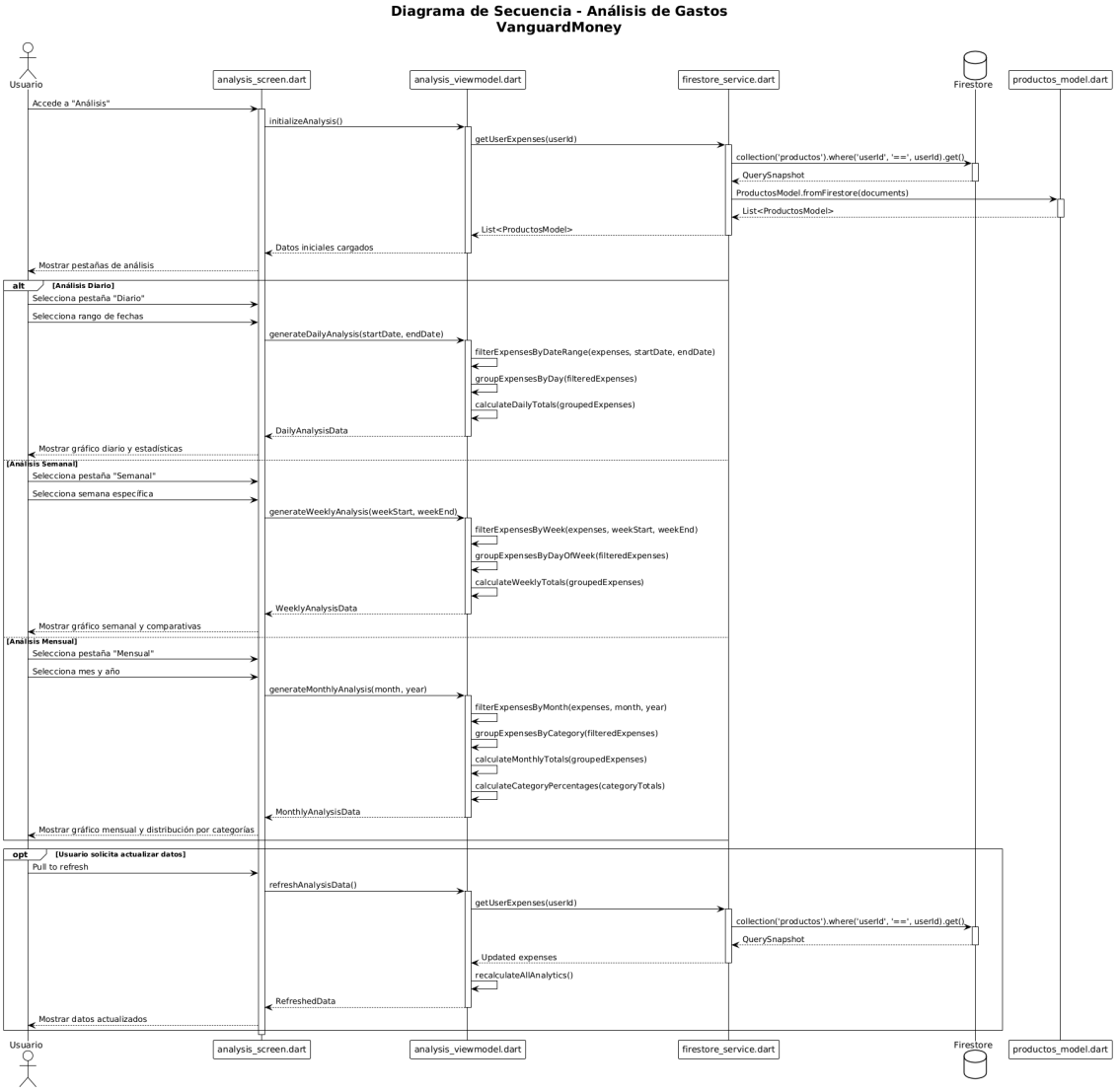
## 2. Secuencia de Escaneo con IA




## 3. Secuencia de Gestión de Planes Financieros



# 4. Secuencia de Análisis de Gastos



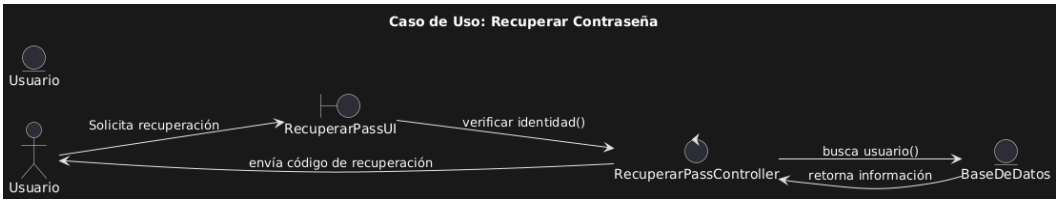
### 3.1.3. Diagrama de Colaboración (vista de diseño)

### 3.1.4. Diagrama de Objetos

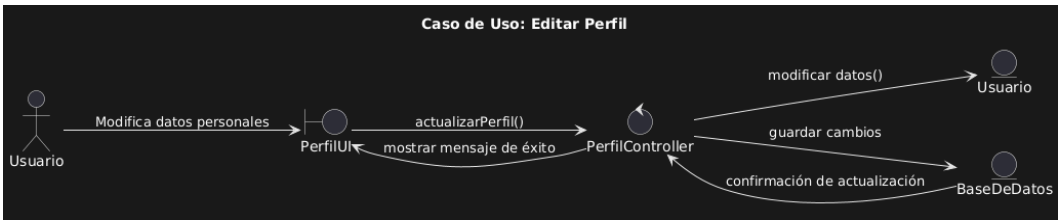
1) Iniciar Sesión


2) Registrar Usuario

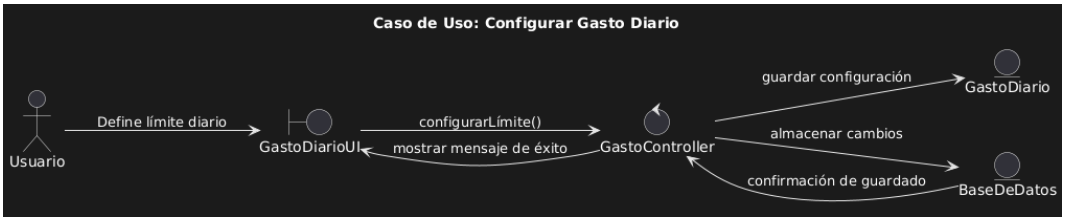

3) Recuperar Contraseña



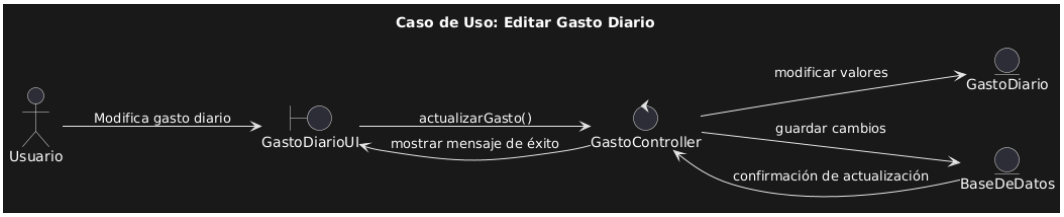

4) Editar Perfil



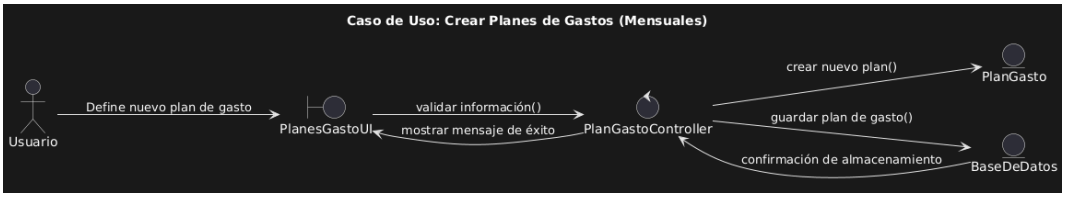

5) Configurar Gasto Diario



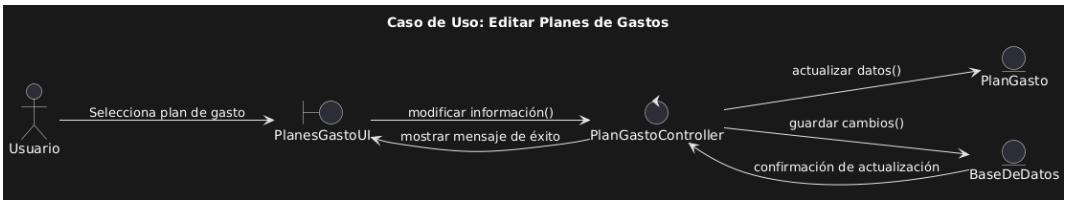
6) Editar Gasto Diario



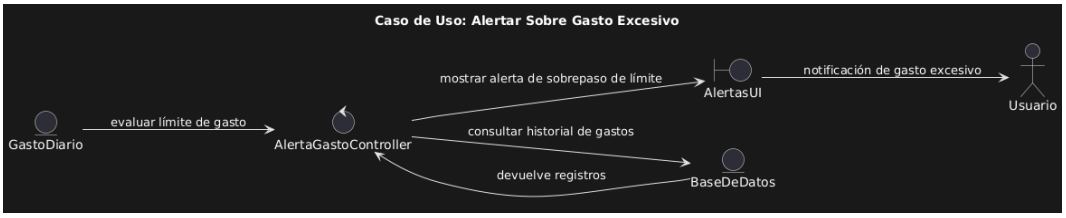

7) Crear Planes de Gastos (Mensuales)




8) Editar Planes



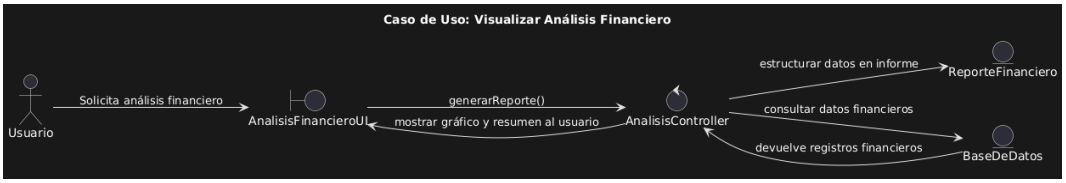

9) Alertar Sobre Gasto Excesivo



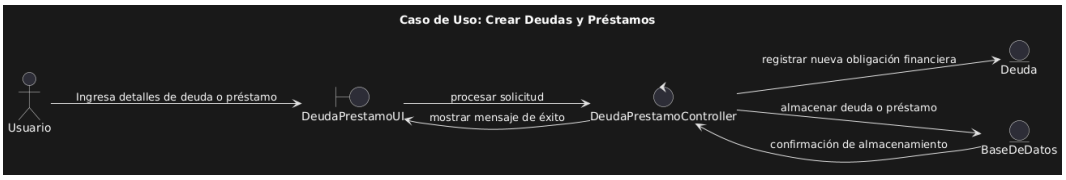

10) Registrar gasto manualmente


11) Registrar gasto por escaneo

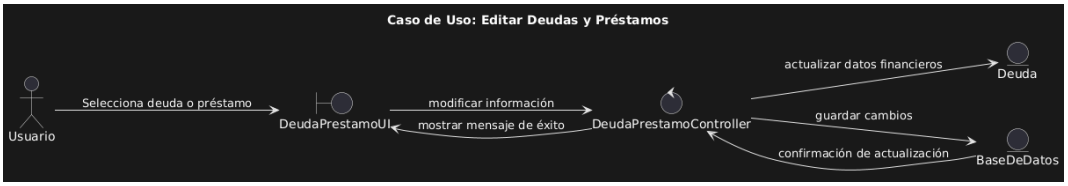
12) Visualizar Análisis Financiero



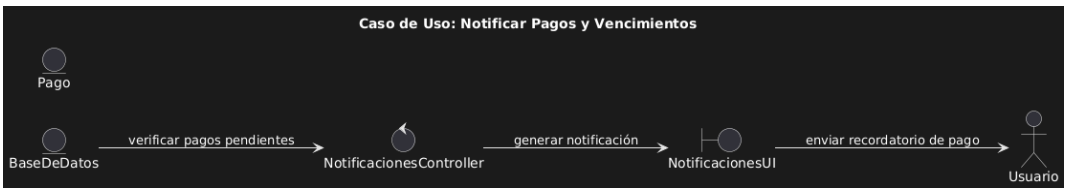

13) Crear Deudas y Préstamos




14) Editar Deudas y Préstamos

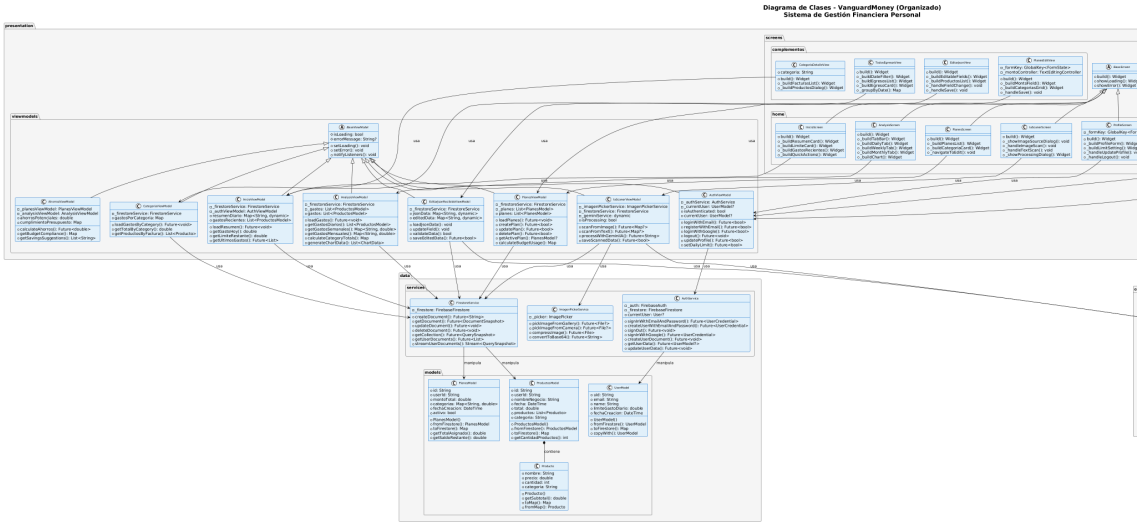



15) Notificar Pagos y Vencimientos

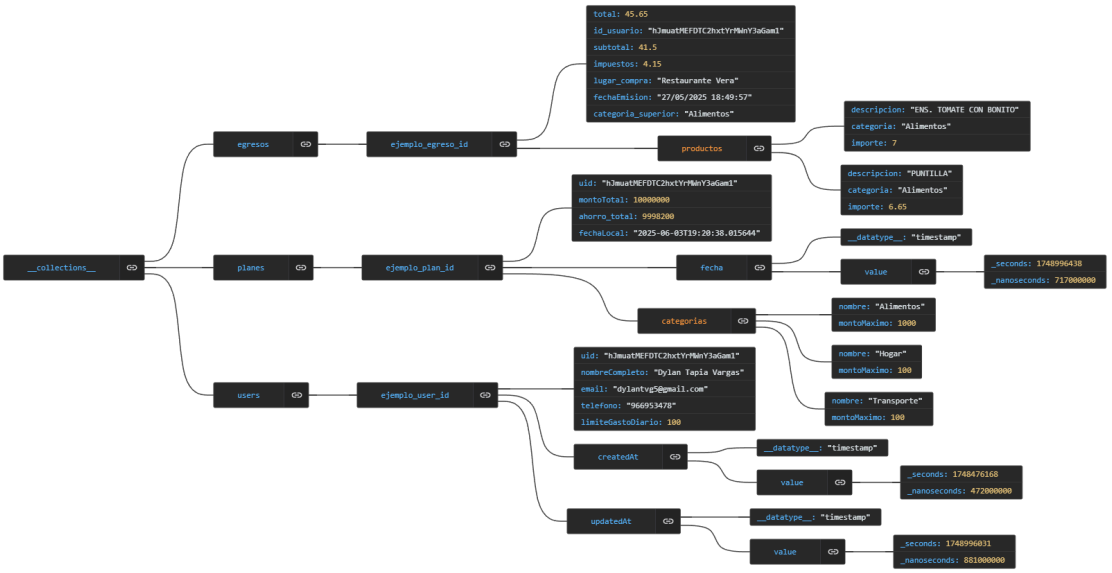



16) Exportar Reportes y Movimientos

### 3.1.5. Diagrama de Clases

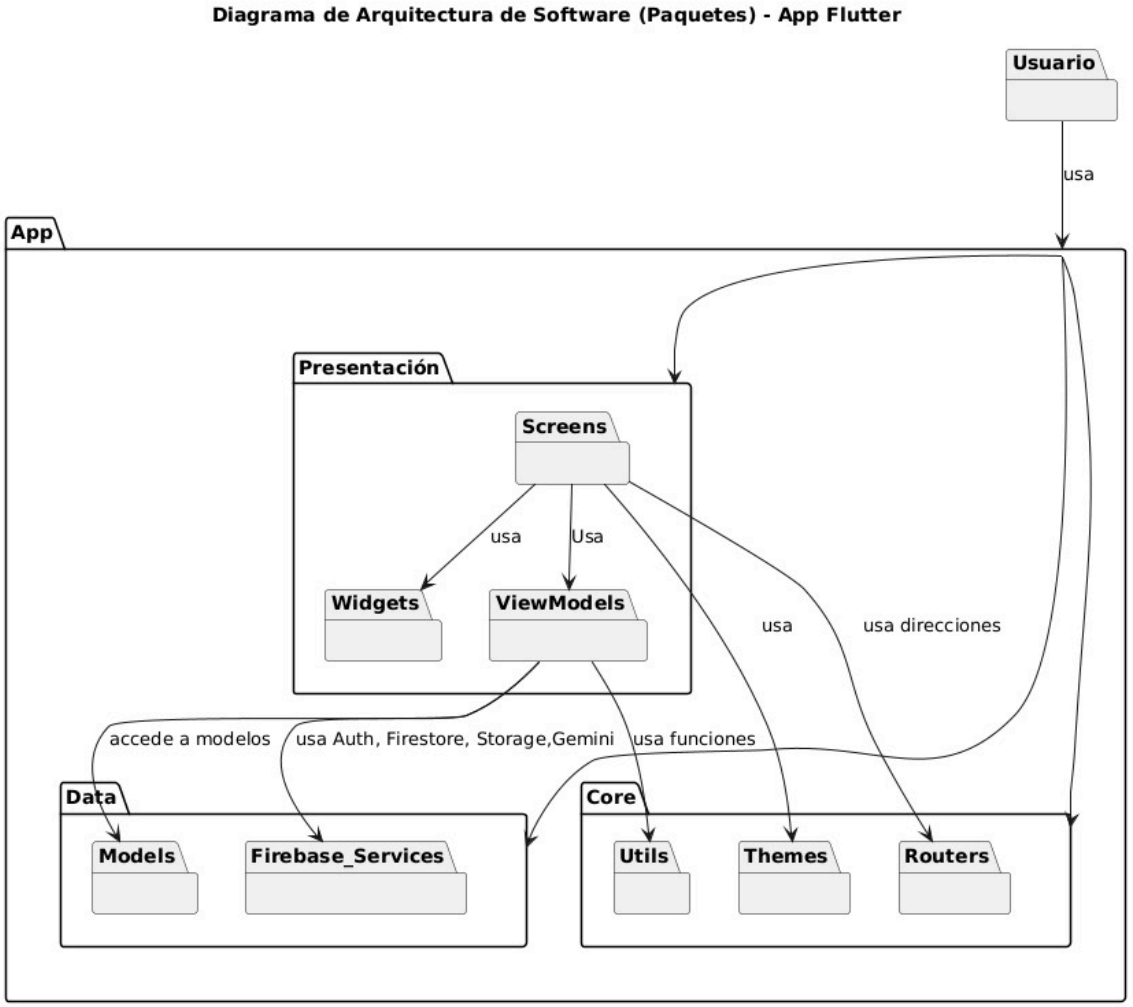



### 3.1.6. Diagrama de Base de datos (relacional o no relacional)

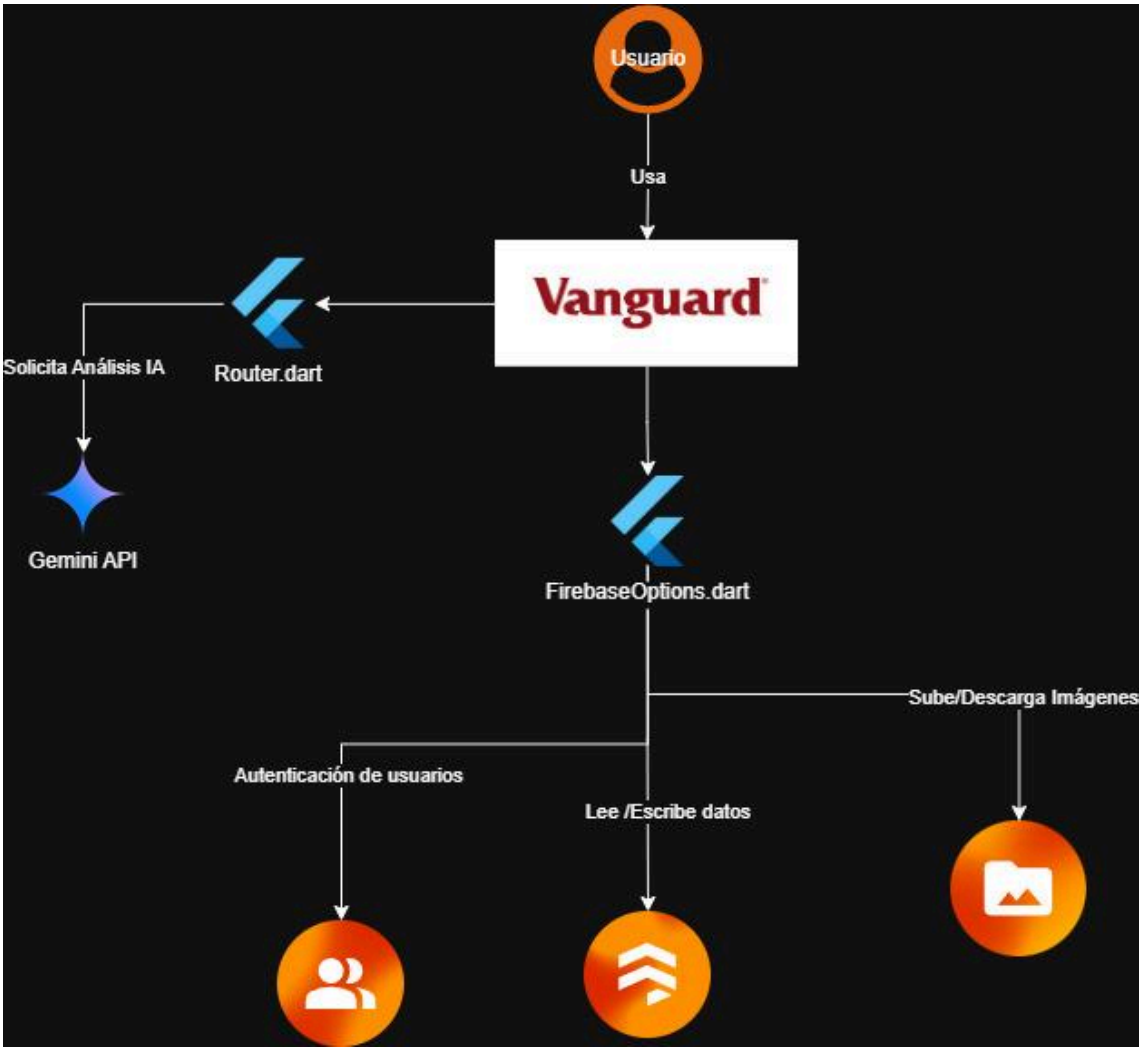


## 3.2. Vista de Implementación (vista de desarrollo)

### 3.2.1. Diagrama de arquitectura software (paquetes)

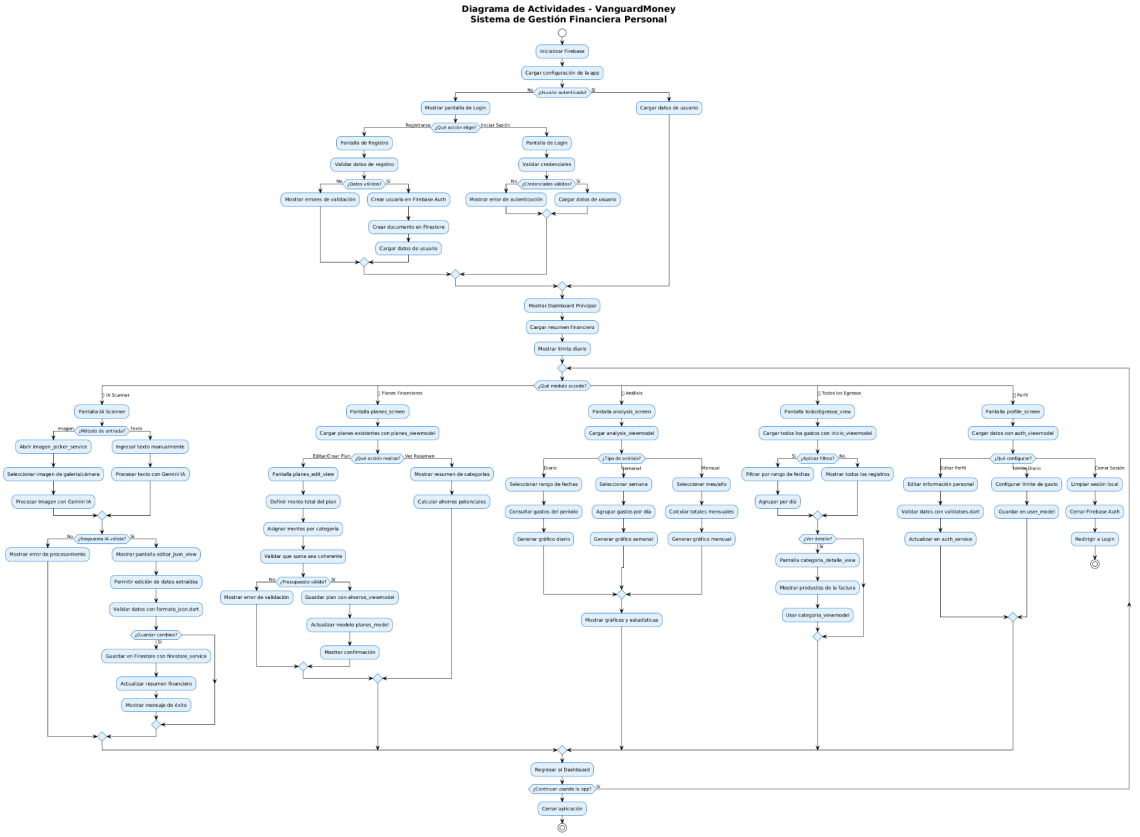


### 3.2.2. Diagrama de arquitectura del sistema



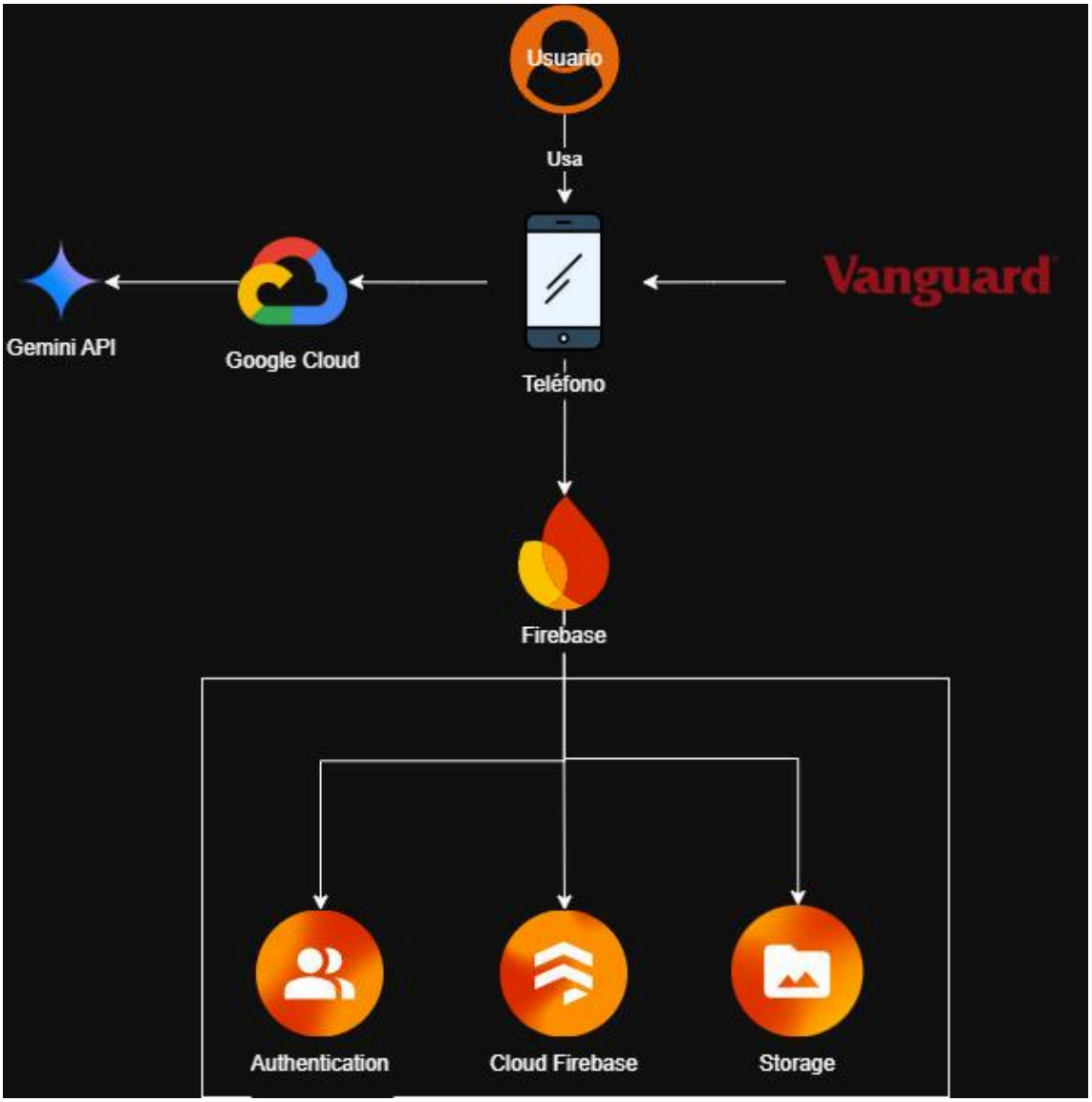
## 3.3. Vista de procesos

### 3.3.1. Diagrama de Procesos del sistema (diagrama de actividad)



## 3.4. Vista de Despliegue (vista física)

### 3.4.1. Diagrama de despliegue



# 4. ATRIBUTOS DE CALIDAD DEL SOFTWARE

Los Atributos de Calidad (QAs) son esenciales para evaluar y garantizar el correcto funcionamiento del sistema, su mantenibilidad, y su capacidad de respuesta ante necesidades reales. En el caso del proyecto Vanguard Money, una solución de gestión financiera personal basada en Flutter, con Firebase como backend y Gemini API para asistencia inteligente, se han definido los siguientes escenarios de calidad:

## Escenario de Funcionalidad

Se refiere a la capacidad del sistema para cumplir los requerimientos del usuario, como el seguimiento de gastos, la gestión de presupuestos, y la asistencia mediante IA (Gemini).

Criterios de evaluación:

- El sistema debe permitir registrar, categorizar y visualizar ingresos y egresos con precisión.
- La API de Gemini debe generar sugerencias útiles basadas en los hábitos financieros del usuario.
- Debe ofrecer alertas automatizadas sobre límites de gasto y presupuestos.
- Las funciones deben estar completamente integradas con Firebase Firestore para almacenamiento en la nube.

## Escenario de Usabilidad

La aplicación debe ser intuitiva para cualquier usuario, incluso sin conocimientos financieros. Debe ofrecer una experiencia fluida, tanto en dispositivos móviles como tablets.

Criterios de evaluación:

- Diseño UI/UX responsivo, accesible y adaptado a diferentes tamaños de pantalla.
- Las acciones principales (agregar gasto, visualizar resumen, recibir consejos) deben ser alcanzables en menos de 3 toques.
- El tiempo de carga de cada vista debe ser menor a 2 segundos.
- Personalización por usuario, por ejemplo, categorías favoritas o temas de color.

## Escenario de confiabilidad

El sistema debe ser seguro, estable y tolerante a fallos, protegiendo la información financiera personal del usuario.

Criterios de evaluación:

- Autenticación segura mediante Firebase Authentication.
- Encriptación de datos sensibles.

- Sincronización consistente con Firestore, incluso en caso de desconexión temporal.
- Disponibilidad mínima del 99%, considerando que puede ser usada en cualquier momento del día.

### Escenario de rendimiento

El sistema debe ser capaz de manejar múltiples operaciones en tiempo real, como sincronización de datos, consultas inteligentes y generación de reportes.

Criterios de evaluación:

- Generación de reportes financieros en menos de 3 segundos.
- Integración fluida con la API de Gemini sin retardos perceptibles.
- Optimización de llamadas a Firebase para evitar exceso de lecturas/escrituras.
- Fluidez en dispositivos de gama media y baja.

### Escenario de mantenibilidad

El sistema debe permitir ajustes rápidos ante nuevos requerimientos, mejoras de seguridad o ajustes en la IA.

Criterios de evaluación:

- Código modular, usando buenas prácticas en Dart y arquitectura limpia.
- Documentación clara en cada módulo (Flutter, Firebase, Gemini).
- Posibilidad de actualizar algoritmos de recomendación sin afectar la app principal.
- Automatización de pruebas mediante GitHub Actions (Snyk, Semgrep y SonarCloud).

### Escenario de Performance (Carga concurrente y eficiencia)

Además del rendimiento individual, el sistema debe comportarse correctamente ante múltiples usuarios y en escenarios de alta carga.

Criterios de evaluación:

- Soporte para múltiples usuarios concurrentes conectados mediante Firebase.
- Pruebas de carga ejecutadas automáticamente con GitHub Actions.
- Reportes generados eficientemente incluso con 6 meses o más de datos históricos.
- Evaluación continua de seguridad y vulnerabilidades con Snyk y Semgrep.