Universidad Don Bosco

Técnico en Ingeniería en Computación

Desarrollo de Software para Dispositivos Móviles

Ing. Kevin Jimenez

Grupo Teoría DSM441 G01T

Tema:

Proyecto Catedra

Jonathan Alejandro Mendoza Olano	MO223025
Jorge Nahum Mira Flores	MF232232
Carlos Francisco Villalobos Romero	VR230136
Sebastian Donovan Torres Gutierrez	TG230977

Link de Github: [Carislost/DSM_ProyectoCatedra01: Proyecto de Catedra fase 01 (github.com)]

Campus Antiguo Cuscatlán, 02 de septiembre de 2024

índice

Introducción

En la era digital actual, la gestión eficiente de información personal y laboral representa un desafío constante para muchas personas. Con un aumento significativo en las responsabilidades diarias, surge la necesidad de herramientas que no solo faciliten la organización de datos, sino que también sean intuitivas y accesibles. En este contexto, el proyecto "Clear Note" emerge como una solución innovadora que aborda estas necesidades de manera práctica y eficaz.

Este documento proporcionará una visión detallada de los objetivos, estrategias y metas específicas para el desarrollo de "Clear Note", una aplicación móvil diseñada específicamente para permitir a los usuarios crear, editar y gestionar diferentes tipos de notas. Con un enfoque prioritario en la eficiencia, la seguridad de los datos y la adaptabilidad, "Clear Note" aspira a revolucionar la manera en que los usuarios interactúan con sus notas, tanto personales como profesionales.

La aplicación propone una plataforma centralizada que integra funcionalidades avanzadas dentro de una interfaz amigable y fácil de usar. A lo largo de este documento, se explorarán en profundidad los aspectos técnicos y operativos involucrados en el desarrollo de la aplicación. Además, se establecerá un marco robusto para la implementación y evaluación de "Clear Note", considerando especialmente el contexto específico de El Salvador.

Objetivos

Objetivo General

Desarrollar una aplicación móvil llamada "Clear Note" que permita a los usuarios crear, editar, y gestionar diferentes tipos de notas de manera eficiente y centralizada, ofreciendo una experiencia de usuario intuitiva y flexible que se adapte a sus necesidades organizativas diarias.

Objetivo Especifico

1. **Implementar un sistema de registro e inicio de sesión de usuarios:** Permitir a los usuarios crear cuentas personales y acceder de manera segura a sus notas desde cualquier dispositivo, asegurando la protección de su información personal.
2. **Desarrollar y optimizar funcionalidades para la creación y gestión de diferentes tipos de notas:** Proporcionar a los usuarios la capacidad de crear notas de tipo lista, donde puedan agregar elementos y marcarlos como completados, así como notas de texto simples, con la posibilidad de añadir más tipos de notas en futuras actualizaciones.
3. **Integrar opciones intuitivas para la edición y eliminación de notas existentes:** Facilitar la organización y actualización de la información almacenada en la aplicación, permitiendo a los usuarios modificar o eliminar sus notas según sea necesario, de manera rápida y sin complicaciones.

Perfil de Proyecto

Nombre de Proyecto: Clear Note

Problematica a Resolver

En la vida cotidiana, muchas personas utilizan diferentes métodos para organizar sus tareas, recordatorios y notas personales, pero a menudo enfrentan dificultades para encontrar una solución centralizada y accesible que les permita gestionar estas necesidades de manera eficiente. Existen aplicaciones especializadas en listas de tareas, blocs de notas, o recordatorios, pero pocas ofrecen la flexibilidad de combinar varios tipos de notas en una sola plataforma. Además, la mayoría de las aplicaciones disponibles carecen de opciones sencillas e intuitivas para registrar, editar y organizar estas notas, lo que resulta en una experiencia de usuario fragmentada y poco satisfactoria.

La aplicación "Clear Note" se propone resolver este problema al ofrecer una plataforma simple y eficiente que permita a los usuarios crear, editar y gestionar diferentes tipos de notas en un solo lugar, con la posibilidad de expandir sus funcionalidades en el futuro para adaptarse a nuevas necesidades de los usuarios.

Metas de la Clear Note

Crear y lanzar una aplicación "Clear Note" que ofrezca una experiencia de usuario sencilla, intuitiva y cómoda, permitiendo a los usuarios gestionar sus notas de diferentes tipos de manera eficiente y sin complicaciones, con un enfoque en la usabilidad y satisfacción del usuario desde la primera versión.

Resultados Esperados de Clear Note

Interfaz de Usuario Intuitiva y Amigable:

- La aplicación "Clear Note" ofrecerá una interfaz de usuario limpia y fácil de navegar, permitiendo a los usuarios crear, editar y eliminar notas con mínima instrucción o confusión.

Gestión Eficiente de Notas:

- Los usuarios podrán gestionar diferentes tipos de notas (listas y notas de texto) de manera eficiente, con funcionalidades que les permitan agregar, tachar, editar y eliminar elementos de sus listas o notas sin complicaciones.

Sincronización Segura de Datos:

- Gracias a la integración con Firebase, los datos de las notas estarán sincronizados en tiempo real y almacenados de forma segura, permitiendo a los usuarios acceder a sus notas desde cualquier dispositivo registrado sin pérdida de información.

Desempeño Rápido y Fiable:

- La aplicación funcionará de manera fluida, con tiempos de respuesta rápidos, tanto al cargar notas como al realizar modificaciones, garantizando una experiencia de usuario satisfactoria.

Adaptabilidad a Futuras Expansiones:

- La arquitectura de la aplicación permitirá la fácil incorporación de nuevos tipos de notas y funcionalidades adicionales en el futuro, en respuesta a la retroalimentación de los usuarios y a las necesidades emergentes.

Alta Satisfacción del Usuario:

- Los usuarios reportarán un alto nivel de satisfacción con la aplicación, valorando su facilidad de uso, la comodidad que ofrece para gestionar diferentes tipos de notas, y la seguridad de sus datos.

Área Geográfica

El desarrollo de la aplicación "Clear Note" se está llevando a cabo en El Salvador. Lo cual influye en varios aspectos del proyecto, como la elección de tecnologías y las consideraciones de conectividad, asegurando que la aplicación sea accesible y relevante para los usuarios dentro del contexto salvadoreño. Además, la proximidad geográfica facilita la colaboración entre los miembros del equipo de desarrollo, que están ubicados en la misma región, permitiendo un trabajo más coordinado y eficiente.

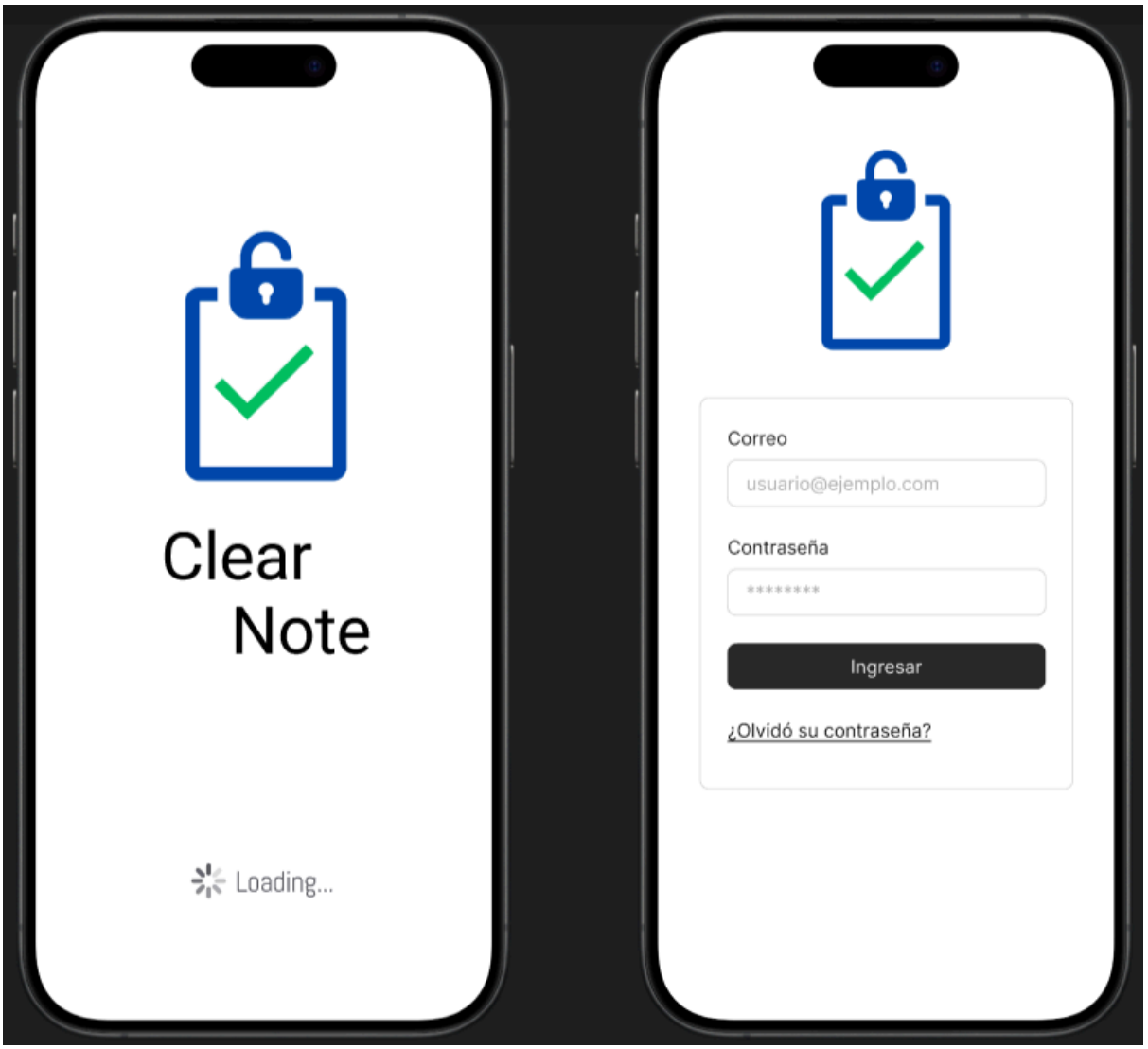
Principales Beneficiarios del Proyecto

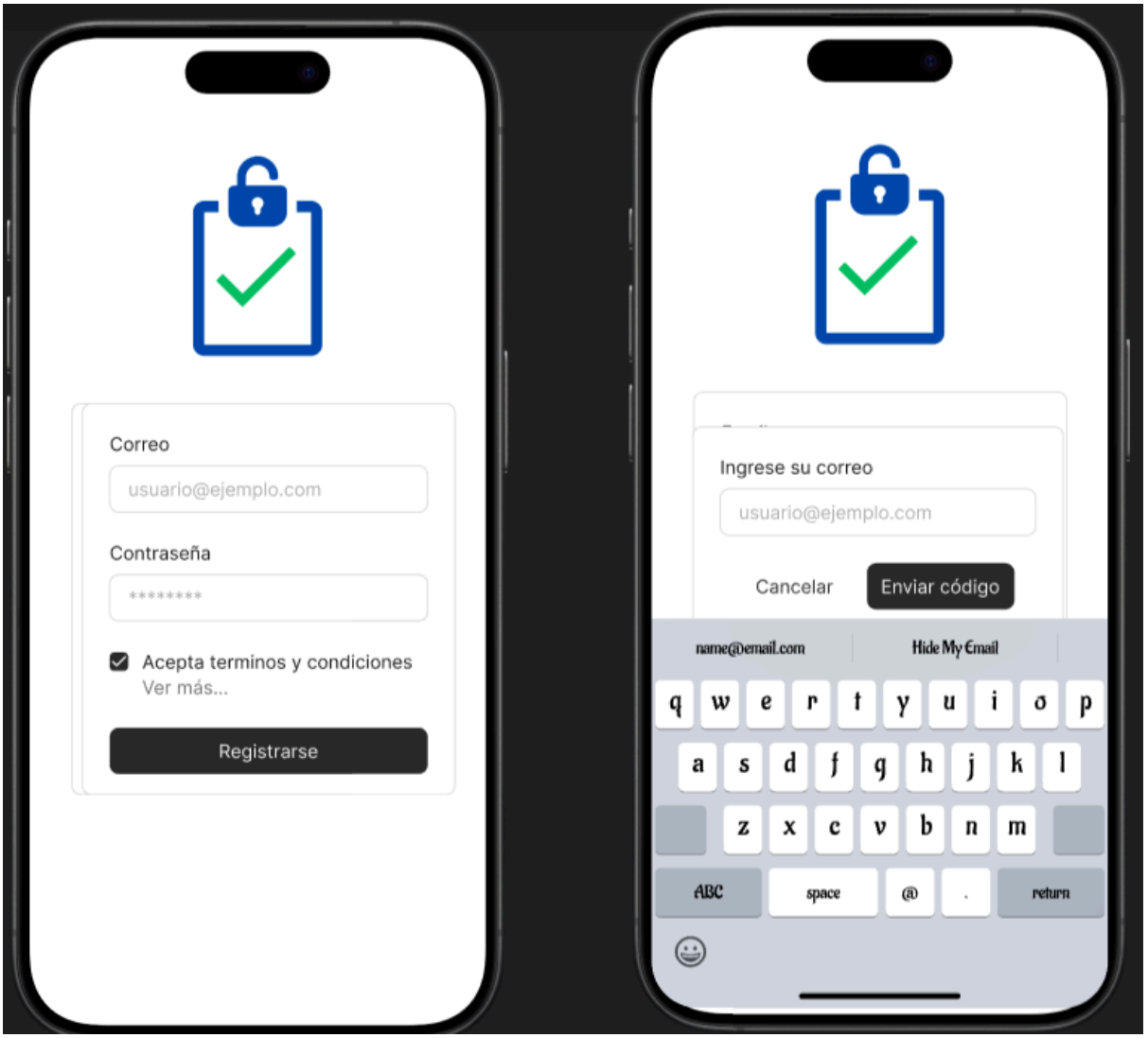
1. **Desarrolladores:**
    - Los desarrolladores que participan en la creación de "Clear Note" serán los principales beneficiarios del proyecto. A través del proceso de desarrollo, ganarán experiencia práctica en programación en Kotlin, uso de Firebase como base de datos, y aplicación de metodologías ágiles como Scrum. Además, este proyecto servirá como un portafolio valioso, demostrando sus habilidades y capacidades en la creación de aplicaciones móviles funcionales.
2. **Usuarios Finales:**
    - Aunque el enfoque principal está en los desarrolladores, los usuarios finales también se beneficiarán del proyecto. Podrán disfrutar de una herramienta eficiente y fácil de usar para gestionar sus notas y tareas diarias, lo que les ayudará a organizarse mejor y ser más productivos.

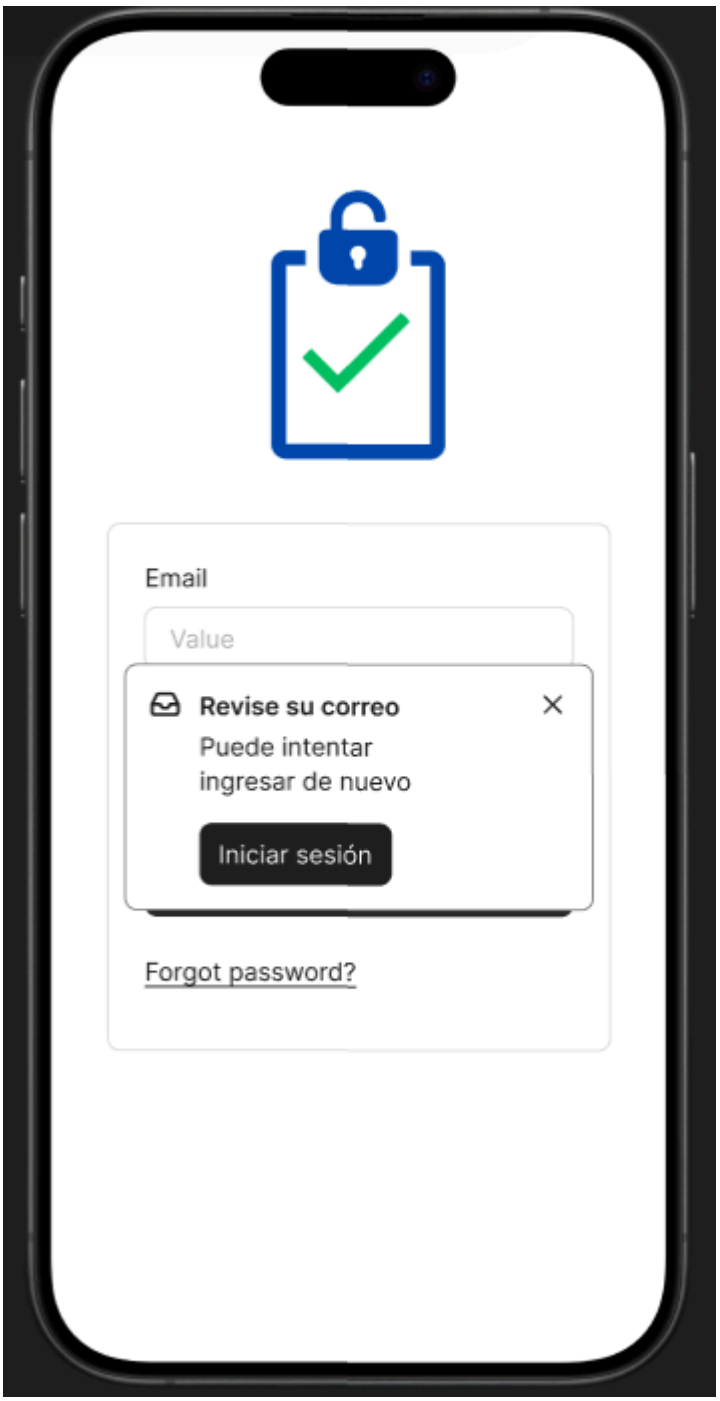
Factores de Éxito para "Clear Note"

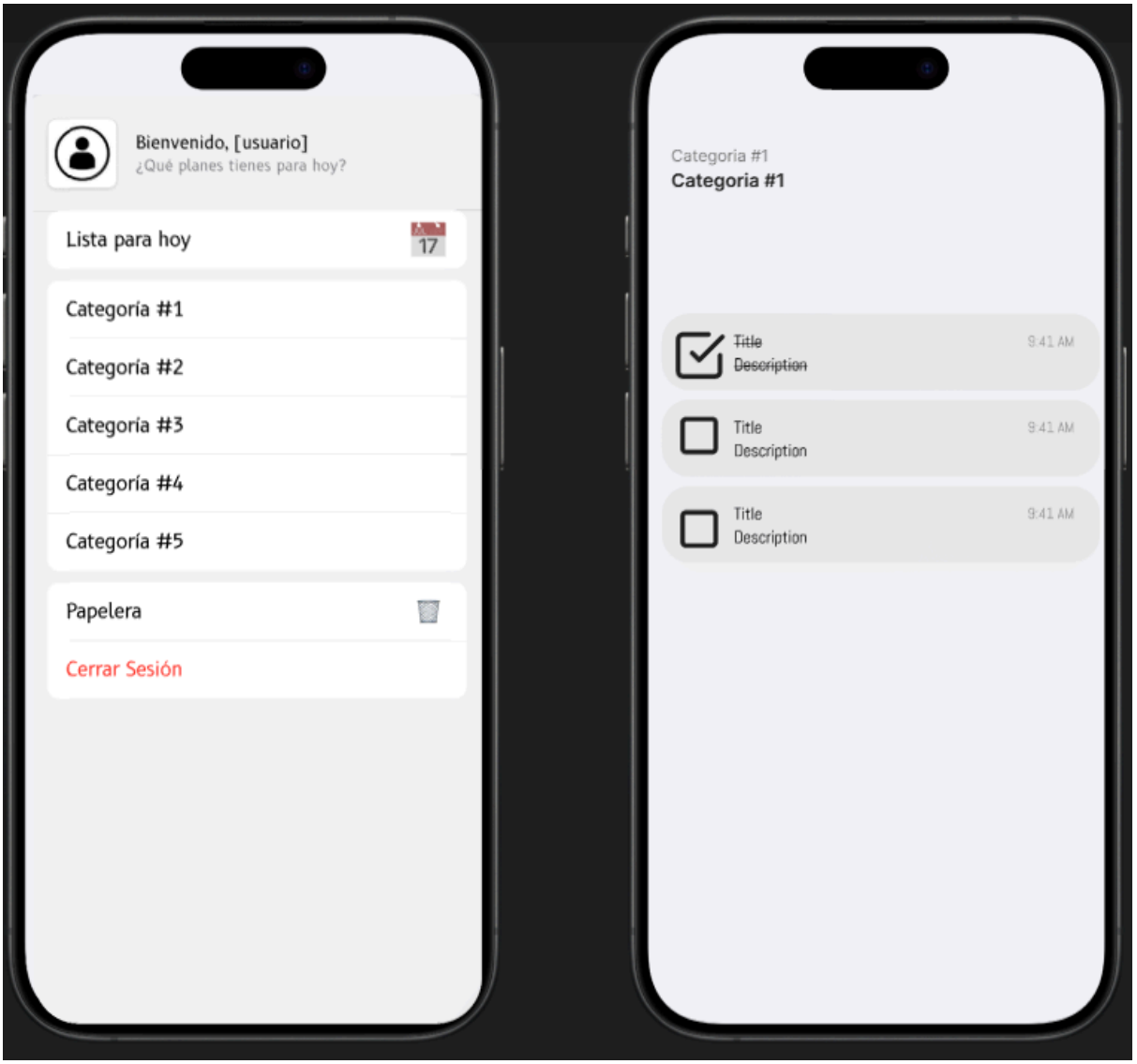
1. **Experiencia de Usuario Intuitiva:**
    - La aplicación debe ofrecer una interfaz de usuario que sea fácil de entender y utilizar desde el primer momento, minimizando la curva de aprendizaje y asegurando que los usuarios puedan realizar sus tareas con rapidez y sin frustraciones.
2. **Estabilidad y Rendimiento:**
    - La aplicación debe ser estable y rápida, con tiempos de carga y respuesta óptimos. El uso de Firebase como base de datos debe garantizar que la sincronización de datos sea fluida y confiable, sin pérdida de información o interrupciones en el servicio.
3. **Desarrollo Ágil y Adaptativo:**
    - La implementación de una metodología ágil como Scrum permitirá al equipo de desarrollo responder rápidamente a los cambios y a la retroalimentación de los usuarios, asegurando que la aplicación evolucione según las necesidades reales y emergentes.
4. **Seguridad de los Datos:**
    - La aplicación debe garantizar la seguridad de los datos de los usuarios, especialmente en lo que respecta a la autenticación y almacenamiento de notas. La implementación de prácticas de seguridad adecuadas, como la autenticación segura y la protección de datos en Firebase, será crucial.
5. **Colaboración Efectiva del Equipo:**
    - La buena comunicación y colaboración entre los miembros del equipo de desarrollo es clave para asegurar que el proyecto avance sin problemas. Las reuniones regulares y la alineación de objetivos ayudarán a que todos trabajen de manera sincronizada hacia un objetivo común.

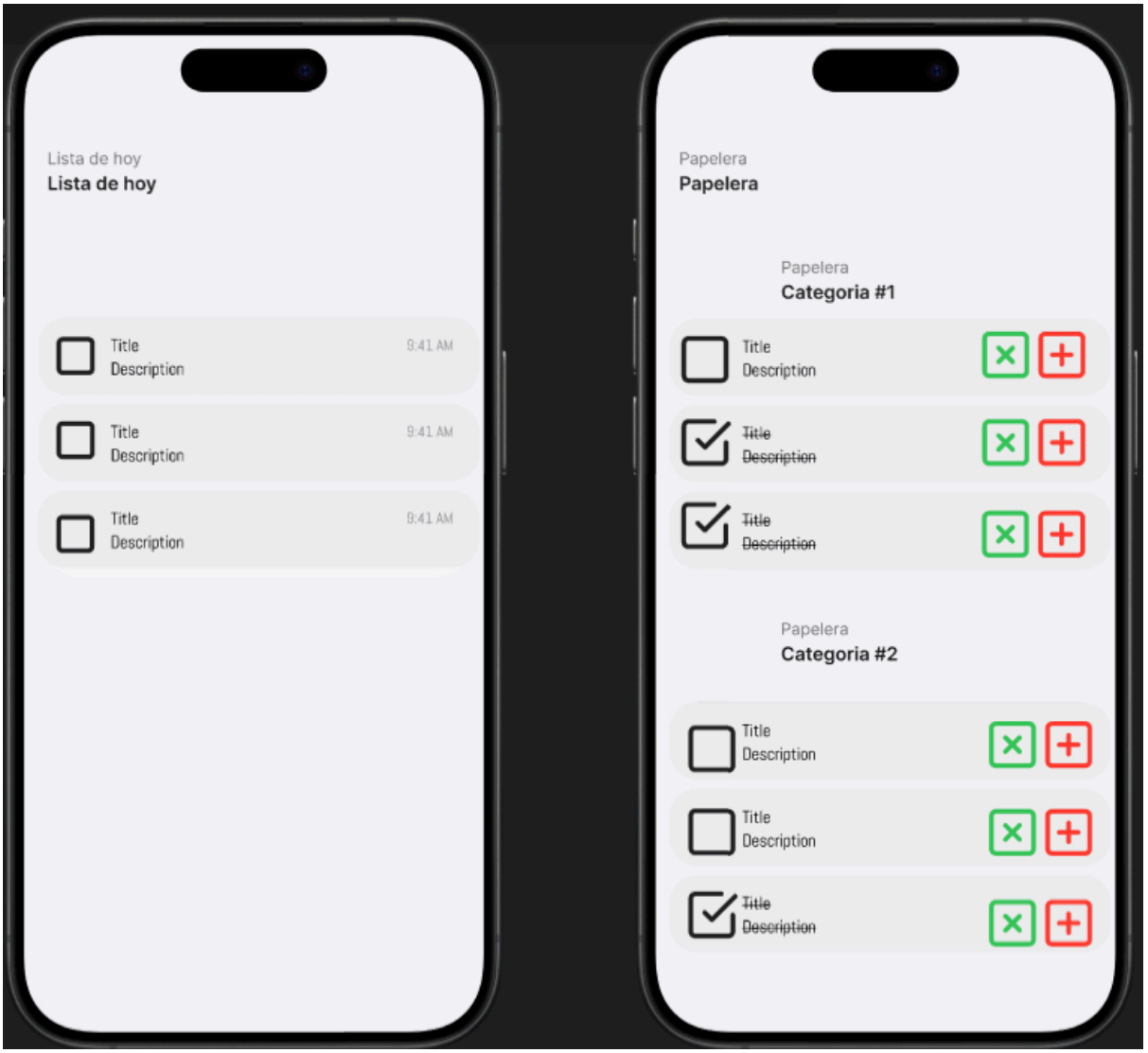
Presentación del diseño UX/UI (mockups)




Correo
usuario@ejemplo.com
Contraseña
****
Acepta terminos y condiciones
Ver más...
Registrarse
Ingrese su correo
usuario@ejemplo.com
Cancelar
Enviar código
name@email.com
Hide My Email
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
ABC
space
@
.
return


Email
Value
Revise su correo
Puede intentar
ingresar de nuevo
Iniciar sesión
Forgot password?


Bienvenido, [usuario]
¿Qué planes tienes para hoy?
Lista para hoy
17
Categoría #1
Categoría #2
Categoría #3
Categoría #4
Categoría #5
Papelera
Cerrar Sesión
Categoria #1
Categoria #1
Title
Description
9:41 AM
Title
Description
9:41 AM
Title
Description
9:41 AM


Lista de hoy
Lista de hoy
Title
Description
9:41 AM
Title
Description
9:41 AM
Title
Description
9:41 AM
Papelera
Papelera
Papelera
Categoria #1
Title
Description
Title
Description
Title
Description
Papelera
Categoria #2
Title
Description
Title
Description
Title
Description

Explicación detallada de la lógica a utilizar para resolver el problema seleccionado, agregar los diagramas UML que sean necesarios.

1. Arquitectura MVVM

MVVM es un patrón de arquitectura que separa la lógica de la aplicación en tres componentes principales:

Model: Representa la fuente de datos de la aplicación, incluyendo la lógica de negocio y las interacciones con la base de datos.
View: La interfaz de usuario que muestra los datos y captura la entrada del usuario.
ViewModel: Actúa como un intermediario entre la View y el Model, gestionando los datos para que la View los consuma.

2. Componentes y Lógica del Proyecto

a) ModelNota
NotaEntity: Esta clase representa una entidad de nota en la aplicación. Para efectos de ejemplificación, imaginamos que contiene atributos como id, title, content, type, isCompleted, y createdAt. Estos atributos nos ayudarían a definir qué información podría tener una nota en la aplicación final.

User (UserEntity): De manera similar, esta clase ejemplifica cómo se podría representar a un usuario en la aplicación. Se plantea que contenga atributos como userId, username, email, y una lista de notas asociadas al usuario.

Repository (RepositoryConcept): Este componente, en términos teóricos, serviría para interactuar con una base de datos o un sistema de almacenamiento de datos. Aquí, actuamos bajo la suposición de que se encargará de manejar las operaciones relacionadas con las notas y los usuarios, proporcionando una capa de abstracción sobre la fuente de datos, ya sea en una base de datos local (como Room) o en la nube (como Firestore).

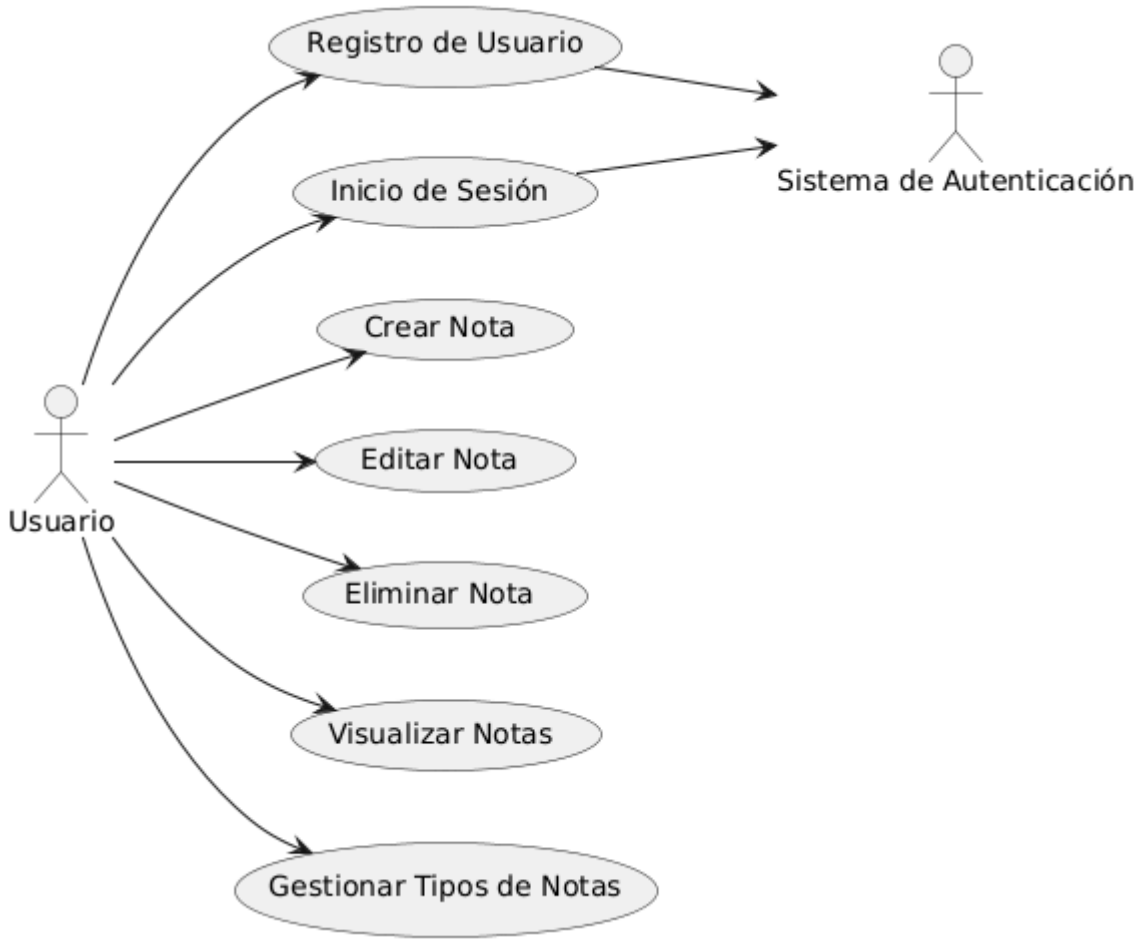
b) ViewModel
NoteViewModel: Contiene la lógica de negocio para gestionar las notas. Se comunica con el Repository para realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar) y proporciona datos a la View.
UserViewModel: Gestiona la autenticación y los datos del usuario.

c) View
MainActivity: La actividad principal que controla la navegación entre las pantallas de inicio de sesión, visualización de notas y creación/edición de notas.
NoteListFragment: Un fragmento que muestra una lista de todas las notas del usuario.
NoteDetailFragment: Un fragmento que permite crear o editar una nota específica.

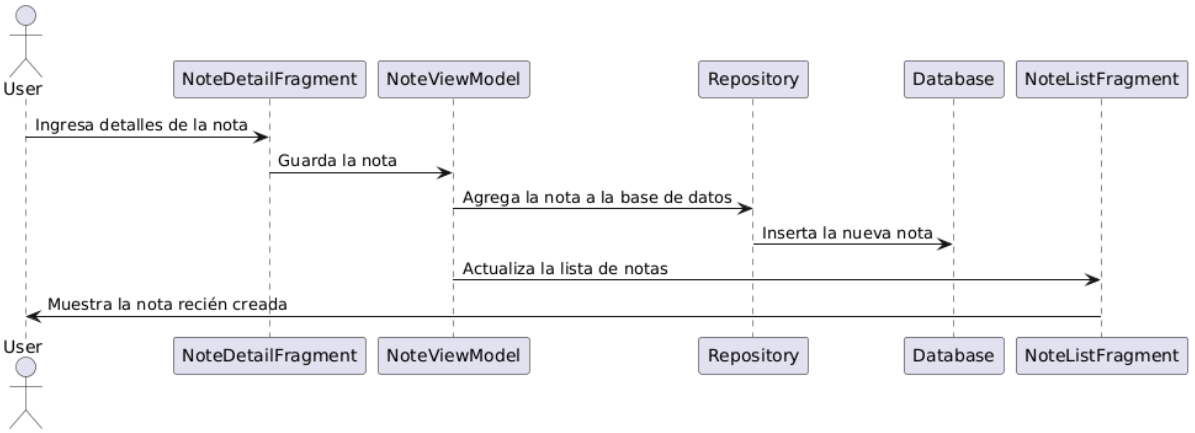
LoginFragment: Fragmento para el inicio de sesión o registro de usuario.

3. Diagrama UML

a) Diagrama de Casos de Uso



b) Diagrama de Secuencia



4. Flujo de Lógica

Inicio de Sesión/Registro:

1) Ingreso de Credenciales:

   En esta fase conceptual, se imagina que el usuario ingresaría sus credenciales (correo electrónico y contraseña) y seleccionaría la opción de iniciar sesión o registrarse.

2) Gestión de Autenticación:

   A nivel teórico, el UserViewModel sería el encargado de manejar la lógica de autenticación. Esto podría incluir la verificación de credenciales contra una base de datos y, si la autenticación es exitosa, la carga de las notas asociadas al usuario.
   Carga de Notas:

   El RepositoryConcept, en un futuro, interactuaría con la base de datos para recuperar todas las notas del usuario una vez autenticado. Este es un paso ejemplificado de cómo se manejarían los datos del usuario.

Visualización de Notas:

1) Consulta de Notas:

   En la interfaz conceptualizada, un NoteListFragment (fragmento de lista de notas) obtendría todas las notas del usuario a través del NoteViewModel. Este ViewModel, hipotéticamente, consultaría al RepositoryConcept para obtener los datos.

2) Verificación de Existencia de Notas:

   Si se determina que el usuario no tiene notas (teóricamente), se mostraría un botón que permitiría crear una nueva nota. Este flujo sugiere cómo podría ser el comportamiento de la aplicación en este escenario.

Creación/Edición de Notas:

1) Selección de Tipo de Nota:

   Se plantea que el usuario podría elegir entre los tipos de notas disponibles, como una lista o un bloc de notas simple.

2) Interfaz de Detalles de Nota:

   El NoteDetailFragment, conceptualmente, presentaría una interfaz donde el usuario podría ingresar o editar los detalles de la nota seleccionada. Esto muestra cómo podría estructurarse la interfaz para diferentes tipos de notas.

3) Guardado de Notas:

   Una vez que el usuario guarda la nota, el NoteViewModel, en teoría, se encargaría de actualizar los datos en la base de datos a través del RepositoryConcept.

Eliminación de Notas:

1) Selección de Nota para Eliminar:

   En el NoteListFragment, el usuario tendría la opción de seleccionar y eliminar una nota. Este proceso muestra cómo podría gestionarse la eliminación de notas dentro de la aplicación.

2) Gestión de Eliminación:

   El NoteViewModel conceptualmente manejaría la eliminación de la nota a través del RepositoryConcept, que, a su vez, actualizaría la base de datos para reflejar los cambios.

Presentar un diagrama grafico del diseño de la arquitectura de software para la aplicación móvil.

En "Clear Note", hemos diseñado la arquitectura de la aplicación siguiendo el patrón MVVM (Model-View-ViewModel), y este diagrama de arquitectura nos ayuda a entender cómo funcionan todos los componentes de la aplicación y cómo se comunican entre sí a modo de ejemplo .

Capas de la Arquitectura

1. Vista (View):

Esta es la capa que está directamente en contacto con el usuario, lo que ven y con lo que interactúan. En "Clear Note", tenemos varias "pantallas" que forman parte de esta capa:

MainActivity: Es la pantalla principal, la que gestiona la navegación entre las diferentes partes de la aplicación.

NoteListFragment: Es donde el usuario puede ver una lista de todas sus notas.
NoteDetailFragment: Es la pantalla donde el usuario puede crear o editar una nota específica.

LoginFragment: Es la pantalla de inicio de sesión, donde el usuario se registra o inicia sesión en su cuenta.

2. ViewModel:

Esta es la capa intermedia, y actúa como un cerebro que conecta lo que el usuario ve con los datos que maneja la aplicación. Hay dos ViewModels importantes en nuestra aplicación:

NoteViewModel: Es el encargado de gestionar todo lo relacionado con las notas. Por ejemplo, cuando un usuario crea una nueva nota o modifica una existente, este ViewModel se asegura de que todo se haga correctamente.

UserViewModel: Se ocupa de la autenticación y la gestión de los datos del usuario, como verificar que los datos de inicio de sesión sean correctos.

3. Modelo (Model):

Aquí es donde se maneja toda la lógica de negocio y los datos de la aplicación. Piensa en esta capa como el almacén de datos y la lógica que hace que la aplicación funcione.

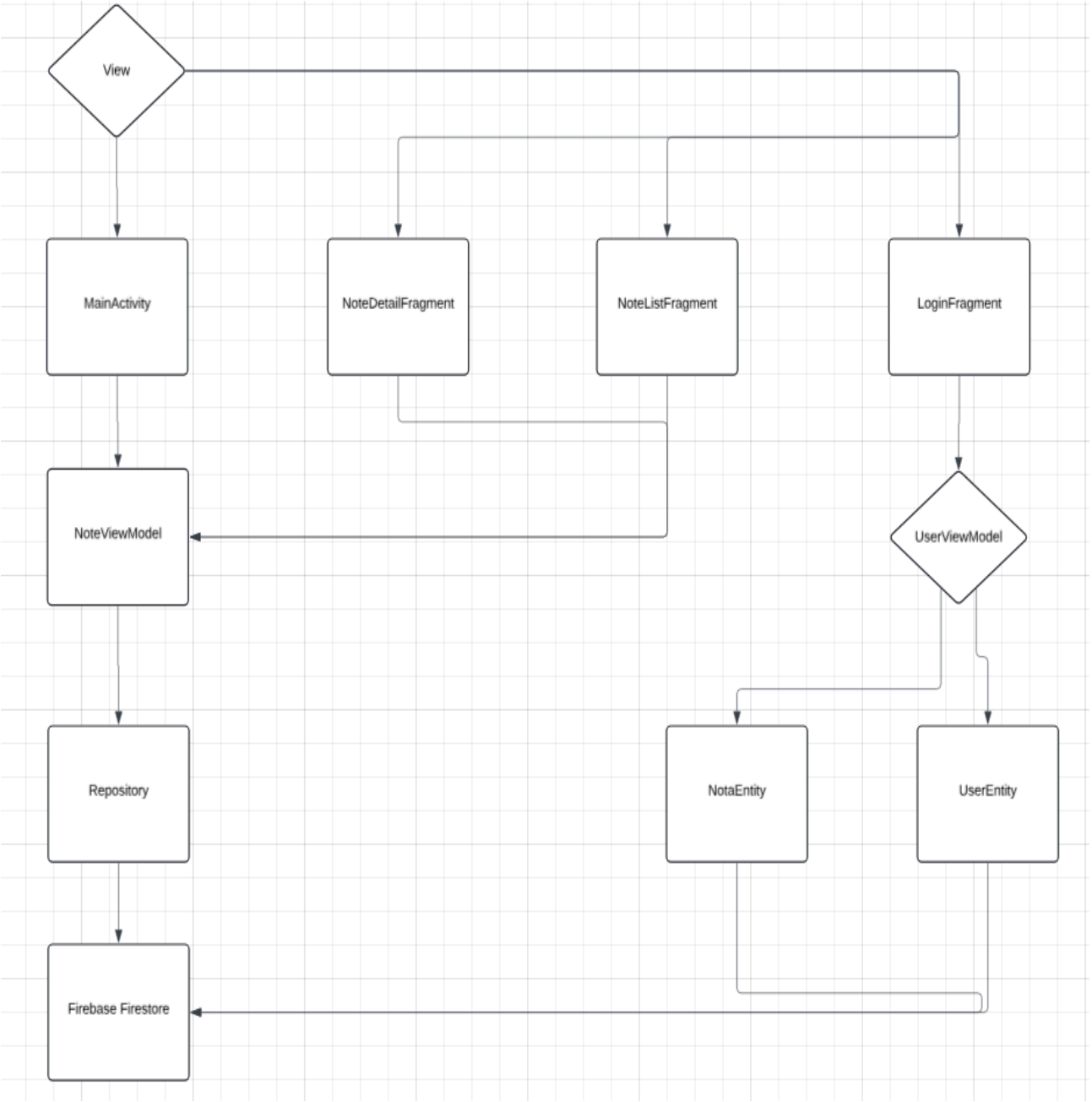
UserEntity y NotaEntity: Estas son las "cajas" que contienen la información sobre los usuarios y las notas. Por ejemplo, una NotaEntity podría tener detalles como el título de la nota, su contenido, y si está marcada como completada o no.
Repository: Este es el "administrador" que se encarga de hablar con la base de datos. Cuando necesitamos guardar una nueva nota o recuperar las notas de un usuario, el Repository es el encargado de hacerlo.

4. Base de Datos (Firebase Firestore):

Finalmente, en la base de datos es donde se almacenan todos nuestros datos. Usamos Firebase Firestore para mantener la información segura y accesible desde cualquier lugar. Esto significa que cuando un usuario guarda una nota en su teléfono, esa nota también se almacena en la nube, lo que permite acceder a ella desde otros dispositivos.

Cómo Interactúan Estas Capas
Cuando un usuario abre "Clear Note" y quiere, por ejemplo, crear una nueva nota:

Primero, la Vista (la pantalla de creación de notas) envía la información al ViewModel.
El ViewModel procesa esa información y la pasa al Repository en la capa del Modelo.
El Repository luego guarda esa nueva nota en Firebase Firestore.
Todo este proceso se hace de manera que el usuario solo vea una interfaz simple y fácil de usar, mientras que en el fondo, todo está organizado y conectado para que funcione sin problemas.


View
MainActivity
NoteDetailFragment
NoteListFragment
LoginFragment
NoteViewModel
UserViewModel
Repository
NotaEntity
UserEntity
Firebase Firestore

Detalle de todas las herramientas a utilizar durante el desarrollo

Aqui daremos una descripción de las herramientas usadas durante el desarrollo de la app Clear Note.

Firebase

Firebase es una plataforma de desarrollo de aplicaciones móviles que ofrece una amplia gama de servicios, incluidos almacenamiento de datos en tiempo real (Firestore) y autenticación de usuarios (Firebase Authentication). Usar Firebase facilita la implementación de una base de datos en la nube y un sistema de autenticación seguro sin necesidad de configurar un servidor backend desde cero, algunas ventajas son Firebase Firestore: Base de datos NoSQL en tiempo real que permite sincronización automática de datos entre dispositivos. Firebase Authentication: Proporciona métodos de autenticación sencillos y seguros (correo electrónico, Google, etc.). Escalabilidad y fiabilidad gestionadas por Google.

Kotlin

Kotlin es el lenguaje de programación oficial para Android. Es moderno, conciso, seguro y compatible con Java. Kotlin permite escribir código más limpio y menos propenso a errores, lo que es ideal para desarrollar aplicaciones como "Clear Note", Algunas ventajas son el código más limpio y legible en comparación con Java. Soporte para funciones avanzadas como coroutines, que facilitan el manejo de tareas asíncronas.

Android Studio

Android Studio es el entorno de desarrollo integrado (IDE) oficial para desarrollar aplicaciones Android. Ofrece un conjunto completo de herramientas para escribir, depurar, probar, y optimizar el código. Además, tiene un excelente soporte para Kotlin, que es el lenguaje de programación que usarás en "Clear Note", Algunas ventajas son la integración con el emulador de Android para pruebas rápidas. Herramientas avanzadas de depuración y análisis de rendimiento. Soporte directo para Gradle, facilitando la gestión de dependencias y el proceso de compilación.

Figma

Figma es una herramienta de diseño colaborativa que permite crear prototipos y diseños de interfaces de usuario (UI/UX). Es perfecta para diseñar los mockups de "Clear Note", ya que permite la colaboración en tiempo real entre los miembros del equipo, algunas ventajas son Colaboración en tiempo real para equipos de diseño y desarrollo. Prototipado interactivo, que permite simular la experiencia del usuario antes de empezar a codificar.

Git y GitHub

Git es un sistema de control de versiones que permite rastrear los cambios en el código a lo largo del tiempo. GitHub es una plataforma basada en la web que facilita la colaboración entre equipos de desarrollo, proporcionando un espacio centralizado para almacenar y gestionar el código fuente, algunas ventajas son Git: Permite colaborar eficientemente, gestionar versiones y revertir cambios si es necesario. GitHub: Facilita la colaboración entre equipos, revisión de código, e integración con herramientas de CI/CD. Integración con Android Studio, lo que permite un flujo de trabajo continuo.

Trello

Esta herramientas de gestión de proyectos te permite organizar tareas, asignar responsabilidades, y seguir el progreso del proyecto de manera visual. Es ideal para la metodología Scrum o Kanban, algunas ventajas son que facilitan la comunicación y la colaboración del equipo, manteniendo todo el trabajo organizado y accesible. Ofrece un tablero Kanban simple y visual para gestionar tareas y sprints.

Postman

Postman es una herramienta para probar APIs, lo que es crucial si tu aplicación interactúa con servicios web o servicios en la nube como Firebase. Te permite realizar solicitudes HTTP y verificar que las respuestas sean las esperadas, algunas ventajas son que es fácil de usar para crear, guardar, y compartir peticiones HTTP. Ayuda a documentar y automatizar pruebas de API. Integración con herramientas de CI/CD para pruebas automatizadas.

Crashlytics

Crashlytics es una herramienta de monitoreo de fallos en tiempo real de Firebase. Es esencial para mantener la calidad de la aplicación después del lanzamiento, ya que te informa sobre fallos y errores que los usuarios están experimentando, algunas ventajas son los informes de fallos en tiempo real con detalles sobre las condiciones que llevaron al fallo. Ayuda a priorizar la resolución de bugs según el impacto en los usuarios. Integración fluida con Firebase y Android Studio.

Presupuesto

1. Costos de Herramientas y Tecnologías:

- **Firebase (Firestore y Authentication):** Firebase es una plataforma robusta y flexible que se encargará de gestionar tanto la base de datos en tiempo real como la autenticación de usuarios. Para la implementación de Firebase, se recomienda utilizar el **Plan Blaze**, que ofrece una estructura de precios basada en el uso real de los servicios. Este plan es ideal para proyectos que están en etapa de crecimiento, ya que permite escalar conforme aumenta la demanda. Se estima que el presupuesto inicial para Firebase, considerando un uso moderado, estaría en el rango de **$50 - $100 mensuales**, dependiendo del tráfico y almacenamiento necesarios para la aplicación "Clear Note".

- **Kotlin y Android Studio:** Kotlin es el lenguaje de programación oficial para el desarrollo de aplicaciones Android, mientras que Android Studio es el entorno de desarrollo integrado (IDE) proporcionado por Google para este fin. Ambas herramientas son completamente gratuitas, lo que significa que no hay costos directos asociados a su uso. Kotlin ofrece un código más conciso y menos propenso a errores, mientras que Android Studio proporciona un conjunto completo de herramientas para escribir, depurar, y optimizar el código de la aplicación.

- **Figma:** Figma es una herramienta de diseño colaborativa que permite a los equipos crear prototipos y diseños de interfaces de usuario (UI/UX) de manera eficiente. Para proyectos pequeños o medianos, la versión gratuita de Figma puede ser suficiente para cubrir las necesidades del equipo. Sin embargo, si se requiere una colaboración más avanzada y acceso a características adicionales, como controles de versiones o colaboración en tiempo real sin limitaciones, es recomendable considerar la suscripción al plan de pago, que cuesta **$12 por usuario al mes**. Esto permitiría un flujo de trabajo más integrado y una mayor productividad en el diseño de la aplicación.

- **GitHub:** GitHub es una plataforma basada en la web para control de versiones, que facilita la colaboración entre los miembros del equipo de desarrollo. Si bien GitHub ofrece un plan gratuito que puede ser adecuado para proyectos públicos, es posible que el equipo necesite un **plan privado** para mantener la confidencialidad del código fuente y de las características en desarrollo. El costo de este plan privado puede variar entre **$4 y $21 por usuario al mes**, dependiendo del nivel de funcionalidades que se requiera, como acceso a repositorios privados, herramientas de integración continua, y más.

- **Trello:** Trello es una herramienta de gestión de proyectos que permite organizar tareas, asignar responsabilidades, y seguir el progreso de manera visual. Para un equipo que sigue metodologías ágiles como Scrum o Kanban, Trello ofrece un entorno sencillo y eficaz. Aunque la versión gratuita de Trello puede ser útil, el **Plan Business Class**, que cuesta **$10 por usuario al mes**, ofrece características avanzadas como integraciones con otras aplicaciones, automatizaciones, y controles de permisos, lo que puede ser beneficioso para mantener el proyecto "Clear Note" bien organizado y alineado con los objetivos del equipo.

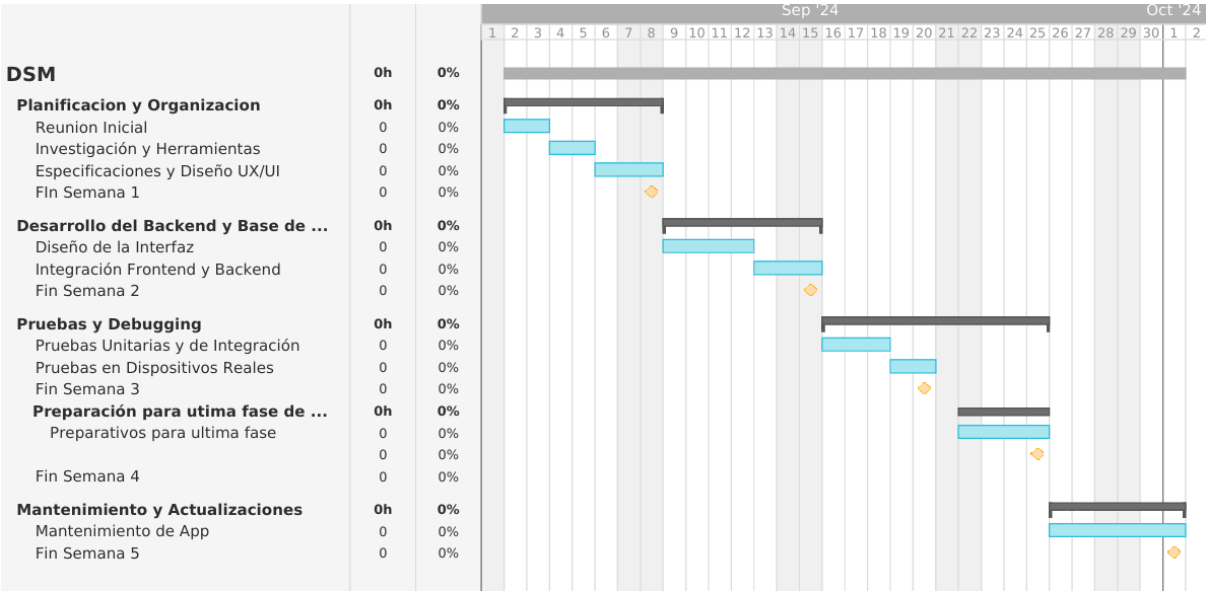
- **Postman:** Postman es una herramienta popular para la creación, prueba, y documentación de APIs, esencial en proyectos que interactúan con servicios en la nube como Firebase. La versión gratuita de Postman es suficientemente poderosa para la mayoría de las necesidades de desarrollo y prueba, permitiendo realizar solicitudes HTTP, testear respuestas, y automatizar pruebas sin ningún costo adicional.

- **Crashlytics:** Crashlytics es un servicio de Firebase que proporciona informes en tiempo real sobre fallos en la aplicación, permitiendo al equipo de desarrollo identificar y corregir errores rápidamente. Dado que Crashlytics está incluido dentro del ecosistema de Firebase, no implica un costo adicional sobre el plan Firebase ya mencionado, lo que lo convierte en una herramienta valiosa y económica para mantener la calidad de la aplicación.

2. Resumen del Presupuesto Inicial:

- **Herramientas y tecnologías:** Considerando todos los elementos mencionados anteriormente, el presupuesto mensual para mantener y operar las herramientas y tecnologías necesarias para el desarrollo y mantenimiento de la aplicación "Clear Note" se estima en un rango de **$100 a $200 al mes**. Este rango incluye los costos asociados con Firebase, Figma, GitHub, y Trello, además de considerar la opción de utilizar versiones pagas para maximizar la eficiencia y seguridad del proyecto.

Cronograma de trabajo



Semana	Día	Actividad	Descripción
Semana 1: Planificación y Organización	1 - 2	Reunión Inicial	Definir el alcance del proyecto, asignar responsabilidades, establecer herramientas de comunicación y seguimiento.
	3 - 4	Investigación y Herramientas	Investigar sobre Firebase, React Native, Figma; Configuración de repositorios y entornos de desarrollo.
	5 - 7	Especificaciones y Diseño UX/UI	Crear bocetos y mockups en Figma/Adobe XD; Definir casos de uso y flujos de usuario.

Semana 2: Desarrollo del Backend, Frontend y Base de Datos	8 - 11	Diseño de la Interfaz	Implementar pantallas principales en React Native; Integración de mockups de diseño con código.
	12 -14	Integración Frontend y Backend	Conectar funcionalidades del frontend con la base de datos; Implementar navegación entre pantallas y gestión de estados con ViewModel.
Semana 3: Pruebas y Debugging	15 - 17	Pruebas Unitarias y de Integración	Escribir y ejecutar pruebas unitarias e integración; Identificación y corrección de bugs.
	18 - 19	Pruebas en Dispositivos Reales	Probar la aplicación en dispositivos Android/iOS; Recopilar feedback y hacer ajustes finales.
Semana 4: Preparacion para ultima fase de proyecto	20 - 24	Preparativos para la última fase de proyecto	Preparar lo que falta para que el proyecto esté en su fase final.
Semana 5: Mantenimiento y Actualizaciones	25 - 30	Mantenimiento de App	Corrección de bugs reportados por usuarios; Implementación de mejoras y nuevas características basadas en feedback.

Bibliografía

- IBM. (s. f.). Desarrollo de aplicaciones móviles. IBM. Recuperado el 2 de septiembre de 2024, de https://www.ibm.com/es-es/topics/mobile-application-development
- FasterCapital. (s. f.). ¿Cuáles son las características esenciales de una aplicación para tomar notas que deberías incluir en tu MVP?. FasterCapital. Recuperado el 2 de septiembre de 2024, de https://fastercapital.com/es/tema/%C2%BFcu%C3%A1les-son-las-caracter%C3%ADsticas-esenciales-de-una-aplicaci%C3%B3n-para-tomar-notas-que-deber%C3%ADas-incluir-en-tu-mvp.html
- Google. (n.d.). *Firebase*. https://firebase.google.com/?hl=es-419
- JetBrains. (n.d.). *Kotlin programming language*. https://kotlinlang.org/
- Google. (n.d.). *Android Studio*. https://developer.android.com/studio?hl=es-419
- Figma. (n.d.). *Figma: Design tool for teams*. https://www.figma.com/es-la/
- Git. (n.d.). *Git*. https://git-scm.com/
- GitHub. (n.d.). *GitHub*. https://github.com/
- Trello. (n.d.). *Trello*. https://trello.com/es
- Postman. (n.d.). *Postman*. https://www.postman.com/
- Firebase. (n.d.). *Firebase Crashlytics*. https://firebase.google.com/products/crashlytics?hl=es-419