**Edushare J.S**

**Integrantes:**

César Francisco Escamilla Heredia

Yarid Alexander Barrientos Magaña

Gracia Lizbeth Aguirre Portillo

Mario de Jesús Rodríguez Méndez

Edwin Steven Valencia Castillo

Erika Stephanie Portillo Gómez

Complejo Educativo Jutta Steiner de Toruño

Modulo 3.2

CALEB VERENICE LOPEZ GUTIERREZ

11 de septiembre del 2024

**Índice**

[Introducción 3](#__RefHeading___Toc649_8220480)

[Formulación del Problema 4](#__RefHeading___Toc651_8220480)

[Planteamiento del problema 6](#__RefHeading___Toc653_8220480)

[Método y técnica de investigación 7](#__RefHeading___Toc655_8220480)

[Requerimientos de creación y utilización del software. 9](#__RefHeading___Toc657_8220480)

[REACT 9](#__RefHeading___Toc659_8220480)

[Estudio de factibilidad. 19](#__RefHeading___Toc661_8220480)

[Modelado de Base de Datos 21](#__RefHeading___Toc663_8220480)

[Modelado Clave-Valor 21](#__RefHeading___Toc665_8220480)

[Modelo documental 22](#__RefHeading___Toc667_8220480)

[Arquitectura de Software 25](#__RefHeading___Toc669_8220480)

[Lógica de negocio/programación 27](#__RefHeading___Toc671_8220480)

[BD Fisica evidencias 31](#__RefHeading___Toc673_8220480)

[Contenido original 36](#__RefHeading___Toc675_8220480)

# **Introducción**

En el presente documento se presenta un análisis detallado y estructurado del proceso de desarrollo de nuestra aplicación, cubriendo el 25% inicial del proyecto. En esta primera etapa, se aborda la **formulación del problema** , donde se identifican las principales dificultades que la aplicación busca resolver. También el **planteamiento del problema** , delimitando los parámetros y características que guiarán el enfoque del proyecto.

El documento también describe **el método y las técnicas de investigación** que se emplearán para fundamentar las decisiones tomadas en esta fase del desarrollo. Posteriormente, se especifican **los requerimientos necesarios para la creación y utilización del software**, destacando el uso de tecnologías como Flutter y React, esenciales para la construcción de la aplicación.

**Finalmente, se realiza un estudio de factibilidad** que evaluará los recursos y las posibles limitaciones, asegurando que este 25% inicial del proyecto sea viable y esté alineado con los objetivos planteados. Este documento tiene como objetivo proporcionar una guía clara y completa para la ejecución exitosa de esta primera fase del desarrollo.

# **Formulación del Problema**

**¿Qué problemas identificamos y qué planeamos solucionar?**

El problema que buscamos solucionar es : **el acceso a recursos educativos confiables y la falta de colaboración efectiva entre estudiantes.**

En particular, los estudiantes de primer y segundo año suelen enfrentar dificultades al realizar tareas, ya sea porque necesitan más material para entender un tema o porque les resulta complicado socializar para obtener ayuda con contenidos o información específica. Queremos proporcionar un espacio de confianza dentro de la misma institución donde puedan encontrar la información que necesitan y colaborar más fácilmente.

Esta idea surge de nuestra propia experiencia como estudiantes. Durante nuestros primeros años, enfrentamos estos mismos desafíos al investigar temas por nuestra cuenta, lo que a menudo nos llevó a sitios web con información de dudosa procedencia, causando confusión. Además, algunos compañeros tenían dificultades para pedir ayuda, incluso a sus propios compañeros , por lo que creemos que un espacio virtual podría hacer que pedir ayuda sea más accesible para ellos.

En ocasiones los profesores no pueden explicar los temas a todos los estudiantes al mismo tiempo, por lo que proporcionar material y dejar algunas explicaciones a través de la app sería muy útil. De esta manera, los estudiantes podrían acceder a la información cuando lo necesiten y aclarar sus dudas.

Nuestra app permitirá que los profesores revisen y respondan preguntas en su tiempo libre, y proporcionen material adicional. Esto también evitaría que los estudiantes repitan las mismas preguntas a los profesores, ya que podrán consultar la app para ver si alguien más ya ha hecho la misma consulta. En resumen, nuestra app busca crear un entorno donde estudiantes y profesores puedan colaborar para ayudarse mutuamente.

Esto proporcionará una mayor **“eficiencia”** en el proceso educativo, al optimizar el acceso a recursos confiables y facilitar la resolución de dudas, permitiendo a estudiantes y profesores gestionar su tiempo de manera más efectiva.

# **Planteamiento del problema**

**¿A qué población va dirigida la app?**

Nuestra aplicación está dirigida principalmente a estudiantes de primer, segundo y tercer año del Complejo Educativo Jutta Steiner de Toruño, enfocándose exclusivamente en materias técnicas. Además, la aplicación permitirá a los profesores gestionar y subir contenido relacionado con las materias que imparten.

**¿Cuál es la muestra?**

La muestra que hemos decidido seleccionar es 70% de los estudiantes de la institución, y de los profesores todos aquello que imparten una materia técnica, excluyendo emprendimiento.

**¿Tamaño de la Muestra ?**

El tamaño de la muestra debería de ser de al menos el 60% de acuerdo o la mayor cantidad al no cumplirse una se realizara otra para poder tener datos de manera real.

.

**Delimitación de la población**

Se delimito y dio prioridad a primer y segundo año, siendo estos años donde más material de apoyo es necesario, o realmente a nosotros nos fue necesario, y a los profesores se delimito de todos los profesores a solo los de las materias técnicas.

# **Método y técnica de investigación**

**Métodos de Investigación Aplicados en su proyecto**

Para obtener información sobre lo necesario para nuestra app se realizará una entrevista y una encuesta la entrevista siendo personalizada solo a profesores a cada uno haciéndole preguntas sobre sus materias técnicas impartidas, y la encuesta para alumnos, con preguntas sobre su preferencia al momento de la lectura y utilización de la app.

**¿Como fue la recolección de información?**

La recolección de información en la entrevista fue realizada de manera personal con cada uno de los profesores, y la encuesta fue hecha de manera virtual con preguntas referente a la preferencia de cada estudiante.

**¿Cuál es el método y técnica de organización?**

El método de organización que nosotros implementamos fue el de Entrevista Semi Estructurada a los profesores y Encuesta Estructurada a los alumnos.

La técnica que se implementó para ambos fue Sistema de Archivos ya que creímos era la más conveniente.

**Análisis de los resultados**

* **ENTREVISTA:** La entrevista nos dio los resultados necesarios para el planteamiento de la base de datos y lo necesario para crear los materiales.
* **ENCUESTA:** La encuesta nos dio datos de la lectura de cada uno de los estudiantes, ya que los estudiantes nos dijeron la preferencia de como cada uno al leer documentos , cada uno de ellos nos dijo con mayor cantidad que preferían leer en modo nocturno o en modo oscuro, prefiriendo colores más opacos y no afecten tanto la vista.

# **Requerimientos de creación y utilización del software.**

## **REACT**

El proyecto "EduShare.js" busca transformar la manera en que las herramientas educativas interactivas son desarrolladas y compartidas, aprovechando el poder de React Native, un framework de código abierto de Facebook que permite crear aplicaciones nativas para múltiples plataformas utilizando una sola base de código en JavaScript. Este documento aborda los requerimientos técnicos y financieros necesarios para la creación, instalación y mantenimiento del entorno de desarrollo utilizando React Native en un sistema Windows. Desde la configuración inicial hasta la implementación de servicios como Firebase, se examinan los pasos esenciales para asegurar que el proyecto EduShare.js funcione de manera eficiente y cumpla con sus objetivos educativos.

**Requerimientos de creacion**

Instalación de React Native: Requerimientos, Dependencias y Configuración del Entorno de Desarrollo en Windows

React Native es un framework de código abierto desarrollado por Facebook que permite la creación de aplicaciones nativas para Android e iOS utilizando JavaScript y React. Su capacidad para compartir gran parte del código entre plataformas lo ha convertido en una opción popular entre los desarrolladores de aplicaciones móviles. Sin embargo, para aprovechar al máximo React Native, es necesario cumplir con ciertos requisitos y configurar adecuadamente el entorno de desarrollo en tu sistema operativo. Este ensayo detalla los pasos necesarios para la instalación de React Native en un entorno Windows, explicando las herramientas asociadas, las dependencias necesarias y cómo configurar adecuadamente el entorno para desarrollar aplicaciones multiplataforma.

**1. Requerimientos del Sistema para Windows**

Para desarrollar aplicaciones con React Native en un sistema Windows, se requiere cumplir con ciertos requisitos del sistema para asegurar que todas las herramientas y procesos funcionen sin problemas:

**Sistema Operativo:** Windows 10 o superior (64 bits). React Native no es compatible con versiones de Windows de 32 bits ni con versiones anteriores a Windows 10. Se recomienda tener instalada la versión más reciente de Windows 10 o Windows 11 para aprovechar las últimas actualizaciones de seguridad y compatibilidad.

**Espacio en Disco**: Al menos 10 GB de espacio disponible para la instalación de Node.js, Android Studio, y otras herramientas necesarias como Visual Studio Code.

**Memoria RAM:** Un mínimo de 8 GB de RAM es recomendado para un desarrollo fluido, aunque 16 GB o más son ideales si planeas ejecutar múltiples instancias de emuladores, Visual Studio Code, y otros programas pesados simultáneamente.

**Procesador:** Procesador Intel Core i3 o superior (o su equivalente en AMD) con soporte para virtualización (para ejecutar emuladores de Android). Un procesador más potente mejorará significativamente la velocidad de compilación y la experiencia general de desarrollo.

**Conexión a Internet:** Es esencial para descargar React Native, herramientas adicionales, y para sincronizar tu entorno de desarrollo con repositorios de código como GitHub. Aunque puedes desarrollar sin conexión, muchas herramientas y dependencias se descargan desde Internet, por lo que una conexión estable es altamente recomendable.

**2. Instalación de React Native y Dependencias en Windows**

**2.1. Instalación de Node.js y npm**

Node.js es un entorno de ejecución para JavaScript, y npm (Node Package Manager) es el administrador de paquetes que viene con Node.js. React Native depende de estas herramientas para la instalación de sus dependencias.

Descargar Node.js:

Visita el sitio oficial de Node.js y descarga la versión recomendada para la mayoría de los usuarios.

Durante la instalación, asegúrate de que la opción "Add to PATH" esté seleccionada para que Node.js y npm sean accesibles desde la terminal.

**2.2. Instalación de React Native CLI**

React Native CLI es la interfaz de línea de comandos oficial para desarrollar aplicaciones con React Native.

Instalación:

Una vez que Node.js y npm estén instalados, abre la terminal y ejecuta el siguiente comando para instalar React Native CLI globalmente: npm install -g react-native-cli

**2.3. Instalación de Android Studio y SDK**

Android Studio es el IDE oficial para el desarrollo de aplicaciones Android y es necesario para la instalación de las herramientas de línea de comando de Android y para configurar emuladores Android.

**Instalación de Android Studio:**

Descarga Android Studio desde su sitio oficial.

Durante la instalación, asegúrate de incluir Android SDK, Android SDK Platform-Tools, y Android Virtual Device.

Configuración del SDK de Android:

Abre Android Studio y navega a SDK Manager (Configuración > Apariencia y comportamiento > Configuración del sistema > Android SDK).

Instala el SDK de Android más reciente y asegúrate de que la opción "Android SDK Command-line Tools" esté seleccionada.

Configuración del Emulador:

Abre AVD Manager en Android Studio (Herramientas > AVD Manager).

Crea un nuevo dispositivo virtual seleccionando un modelo de teléfono Android y una imagen del sistema adecuada.

**2.4. Instalación de Visual Studio Code y Extensiones**

Visual Studio Code (VS Code) es un editor de código que, con las extensiones adecuadas, es ideal para desarrollar aplicaciones React Native.

**Instalación de VS Code:**

Descarga e instala Visual Studio Code desde su sitio oficial.

Instalación de Extensiones para React Native:

Abre VS Code y navega al Marketplace de Extensiones (Ctrl+Shift+X).

Busca e instala las extensiones "React Native Tools" y "ES7+ React/Redux/React-Native snippets" para facilitar el desarrollo con React Native.

**3. Configuración de Variables de Entorno**

Es importante asegurarte de que todas las herramientas estén correctamente configuradas en el sistema para evitar problemas durante el desarrollo.

Node.js: C:\Program Files\nodejs\

Android SDK: C:\Users<tu-usuario>\AppData\Local\Android\Sdk\platform-tools

Para verificar que todas las herramientas estén correctamente configuradas, ejecuta el comando react-native doctor en la terminal. Este comando revisará la configuración del entorno de React Native y proporcionará instrucciones para corregir cualquier problema.

**Recursos Operativos(Equipo de desarrollo)**

El equipo de desarrollo de EduShare.js está compuesto por profesionales especializados en diversas áreas clave para el éxito del proyecto. Cada integrante desempeña un rol fundamental, aportando su experiencia y habilidades para llevar a cabo las diferentes etapas del desarrollo.

- **Analista:**

- Gracia Lizbeth Aguirre Portillo: Es la encargada de analizar los requisitos del proyecto, traducir las necesidades de los usuarios en especificaciones técnicas, y garantizar que el desarrollo cumpla con los objetivos establecidos. Su papel es crucial para la planificación y la definición de las funcionalidades de EduShare.js.

**- Programadores:**

- Yarid Alexander Barrientos Magaña: Responsable del desarrollo y programación de las funcionalidades principales de EduShare.js. Su labor se centra en escribir el código necesario para que la aplicación funcione correctamente en las plataformas objetivo.

- Erika Stephanie Portillo Gómez: Trabaja en conjunto con Yarid en la implementación del código, asegurándose de que las funcionalidades sean robustas, eficientes y que cumplan con los estándares de calidad requeridos.

**- Diseño de Interfaz:**

- Mario de Jesús Rodríguez Méndez: Encargado del diseño de la interfaz de usuario (UI), se enfoca en crear una experiencia de usuario (UX) intuitiva y atractiva. Mario se asegura de que la aplicación no solo sea funcional, sino también visualmente coherente y fácil de usar.

**Base de Datos**

- Edwin Steven Valencia Castillo: Gestiona y mantiene la base de datos del proyecto, asegurando la integridad y seguridad de los datos. Su trabajo es fundamental para el almacenamiento eficiente y la recuperación de la información necesaria para el funcionamiento de EduShare.js.

**Diseño de Base de Datos**

- César Francisco Escamilla Heredia: Se encarga del diseño y estructura de la base de datos, definiendo las relaciones y las entidades necesarias para almacenar los datos de manera óptima. Su rol es esencial para garantizar que la base de datos soporte todas las operaciones requeridas por la aplicación sin problemas de rendimiento.

**Recursos de utilizacion del software.**

**Recursos Financieros para Mantener el Proyecto EduShare.js en React Native**

EduShare.js es un proyecto innovador desarrollado en React Native que tiene como objetivo proporcionar herramientas educativas interactivas y compartir conocimientos de manera eficiente. Para garantizar que este proyecto funcione sin interrupciones y continúe brindando valor a sus usuarios, es crucial contar con los recursos financieros necesarios para cubrir los costos básicos mensuales.

**Costos de Energía.** El proyecto requiere un flujo continuo de energía eléctrica para mantener los servidores, equipos de desarrollo y dispositivos conectados y funcionando correctamente. Se estima que el gasto mensual en energía es de aproximadamente 20 a 30 dólares. Este costo cubre el consumo eléctrico necesario para operar las máquinas de desarrollo, servidores, y otros dispositivos electrónicos esenciales.

**Costos de Internet**. El acceso a una conexión a internet confiable y de alta velocidad es fundamental para el desarrollo y la operación diaria de EduShare.js. Tanto para la comunicación con colaboradores como para la actualización y despliegue de aplicaciones, es indispensable mantener una conexión constante. El costo mensual estimado para un plan de internet adecuado es de 30 dólares.

**Otros Gastos Adicionales**. Aunque los principales costos mensuales son la energía y el internet, es importante considerar otros gastos que puedan surgir, como mantenimiento de hardware, suscripciones a servicios en la nube o herramientas de desarrollo, y la inversión en marketing para promover el proyecto.

**Recursos Tecnológicos para Mantener el Proyecto EduShare.js en React Native**

EduShare.js es una plataforma desarrollada en React Native que busca ofrecer herramientas educativas interactivas y una experiencia de usuario fluida. Para asegurar su funcionamiento adecuado, es vital contar con los recursos tecnológicos necesarios. A continuación, se describen los principales recursos tecnológicos requeridos para el proyecto:

**Conexión a Internet**. Una conexión a internet estable y de alta velocidad es esencial para mantener EduShare.js operativo. El acceso a internet es fundamental tanto para el desarrollo y actualización del proyecto como para el correcto funcionamiento de la aplicación en tiempo real, especialmente cuando se sincroniza con servicios en la nube como Firebase. Se recomienda una conexión de al menos 30 Mbps para garantizar una experiencia fluida tanto para los desarrolladores como para los usuarios.

**Energía Eléctrica.** Para asegurar que los servidores, equipos de desarrollo y dispositivos conectados se mantengan operativos, es necesario un suministro constante de energía eléctrica. Esto incluye la energía para mantener los servidores que alojan la aplicación, así como los dispositivos de desarrollo utilizados por el equipo. La estimación de consumo energético mensual puede variar entre 20 a 30 dólares, dependiendo del equipamiento y las horas de uso.

**Memoria RAM.** El desarrollo en React Native, junto con las herramientas de desarrollo como Android Studio o Visual Studio Code, puede ser intensivo en el uso de recursos. Se recomienda contar con dispositivos que tengan al menos 8 GB de RAM, aunque 16 GB de RAM sería ideal para un rendimiento óptimo, especialmente cuando se trabaja con múltiples emuladores y se compilan proyectos grandes. Un buen manejo de la memoria asegura que el desarrollo y las pruebas se realicen sin contratiempos ni demoras.

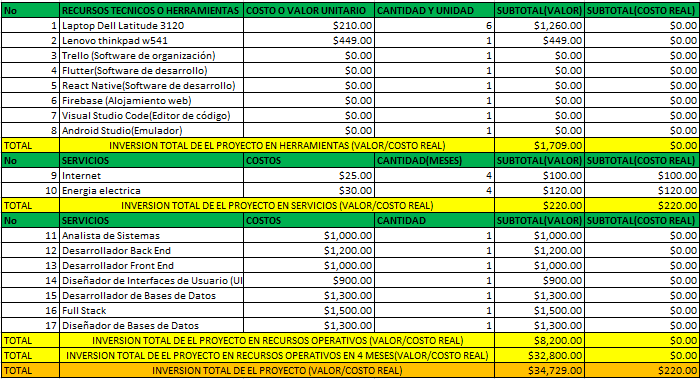
**Firebase**. Es una plataforma integral que ofrece una gama de servicios en la nube que son esenciales para EduShare.js. Firebase no solo proporciona almacenamiento en tiempo real para datos y autenticación de usuarios, sino que también ofrece servicios como Firebase Cloud Messaging para notificaciones, Firebase Analytics para monitoreo y análisis de uso, y Firebase Hosting para desplegar la aplicación.

**Conclusión**

El éxito del proyecto EduShare.js depende de una adecuada configuración del entorno de desarrollo en React Native y de la disponibilidad de recursos tecnológicos y financieros. Desde la instalación de React Native en un sistema Windows hasta la implementación de servicios como Firebase, cada paso es crucial para garantizar un desarrollo fluido y una operación continua. Además, asegurar la disponibilidad de recursos como energía eléctrica, una conexión a internet estable, y dispositivos con suficiente memoria RAM es esencial para mantener la eficiencia y escalabilidad del proyecto. Con estos elementos en su lugar, EduShare.js está bien posicionado para ofrecer herramientas educativas innovadoras que puedan hacer una diferencia significativa en la comunidad educativa.

# **Estudio de factibilidad.**

**PRESUPUESTO DE LA APLICACIÓN**

Evaluacion del valor tecnico,operativo y financiero (Valor/Costo Real)

# **Modelado de Base de Datos**

A continuación se presentara el modelado de la base de datos del proyecto , este modelado está basado en una BD NoSQL específicamente en firebase por lo que la representación tendrá sus respectivas explicaciones para comprender cada parte.

En nuestro proyecto con firebase se usaran 2 modelados : clave valor para el caso de firebase authenthication , documental en el caso de firestore y se tendrá un sistema de guardado de archivos en Cloud Storage por lo que solo se dará una representación simple de las carpetas además de las reglas que se usaran.

## **Modelado Clave-Valor**

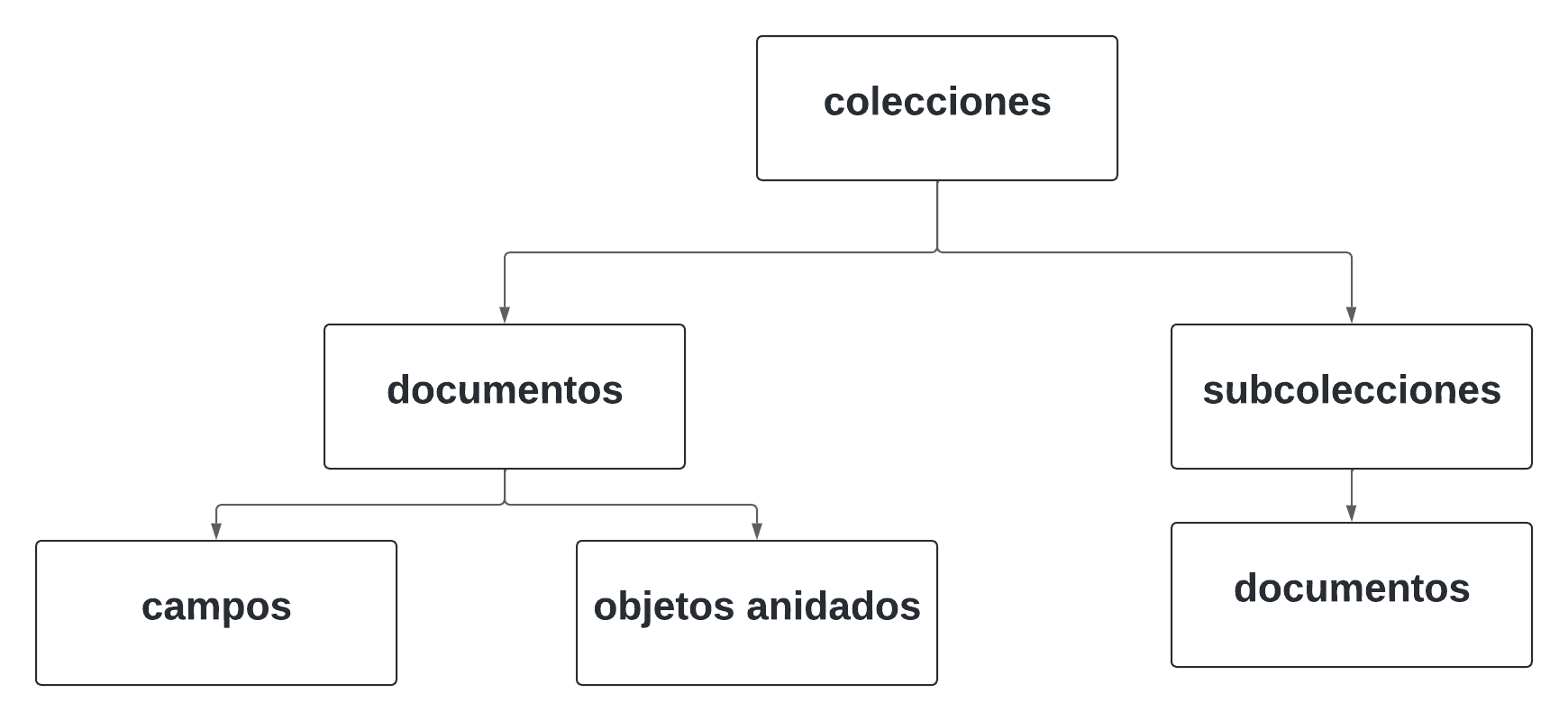
Es como un diccionario de datos gigante donde cada clave tiene un identificador único y cada clave esta asociada a un valor que puede ser cualquier dato . Esto se usa en firebase authentication.

|  |  |
| --- | --- |
| **Clave** | **Valor** |
| Identificador | graciaportillo8@gmail.com |
| UID de usuario | uhgotn9sNySBfv9QlaxaZzaW... |
| Proveedores | Externos como (Facebook,Gmail,Google juegos) |
| Fecha de creación | 6 sept 2024 |
| Fecha de acceso | 6 sept 2024 |

Representación de los datos.

## **Modelo documental**

A continuación se te presenta la estructura del modelo en firestore.



**Colecciones.** Son los contenedores de los documentos.

**Documentos. S**on las unidades de almacenamiento dentro de la colección.

**Campos.** Son los pares clave-valor que contienen los datos dentro del documento. Un campo puede ser una cadena de texto, un número, un booleano hasta una lista.

**Objetos Anidados.** Son estructuras de datos complejas dentro de un documento, que pueden contener varios campos dentro de un solo campo.

**Subcolecciones.** Un documento puede contener una o más subcolecciones, que a su vez contienen más documentos. Esto permite una estructura jerárquica de datos, donde se pueden organizar los datos en partes mas pequeñas y organizadas. Ideal para hacer menos consultas.

**Documentación de la BD en Firestore**

Aquí se explicaran los colecciones , documentos , campos ,subcolecciones , objetos anidados de nuestra BD.

***Users***

Almacena la información de los usuarios.

|  |
| --- |
| **uhgotn9sNySBfv9QlaxaZzaW** |
| {  “displayname”:”Juan ”,  “email”:”juan@gmail.com”,  “profileImage”:”https//firebasestorage.googleapis.com/v0/edushare.js.app/alt=media&token=ijfvniv-rvrovroinv-rvrvr”  “role”:”docente”  } |

***Messages***

Almacena los textos que se envían

|  |
| --- |
| **ws6nk2FVD0bQhnTBD12K** |
| *{*  *“sentBy”:”* “uhgotn9sNySBfv9QlaxaZzaW”,  “sentByName”:”Juan”,  “sentByProfileImage”:”https://firebasestorage.googleapis.com/v0/b/edusharejs.appspot.com/o/predeterminado%2F11.png?alt=media&token=be709679-3df5-4e20-828c-bc043f33e677"  “text”:”hola”,  “timestamp”: 8 de septiembre de 2024, 12:54:29 p.m. UTC-6  } |

***files***

Almacena los archivos que se suben

|  |
| --- |
| **ws6nk2FVD0bQhnTBjiieei** |
| {  “description”:”POO”,  “fileID”:”jicroiuvuirf48894ir989ffbr”,  “uploadedBy”:”uhgotn9sNySBfv9QlaxaZzaW”  “Url”:”https://firebasestorage.googleapis.com/v0/b/edusharejs.appspot.com/o/predeterminado%2F11.png?alt=media&token=be709679-3df5-4e20-828c-bc043f33e677”  } |

***chatRooms***

Sirve para identificar donde se mandaran los chats

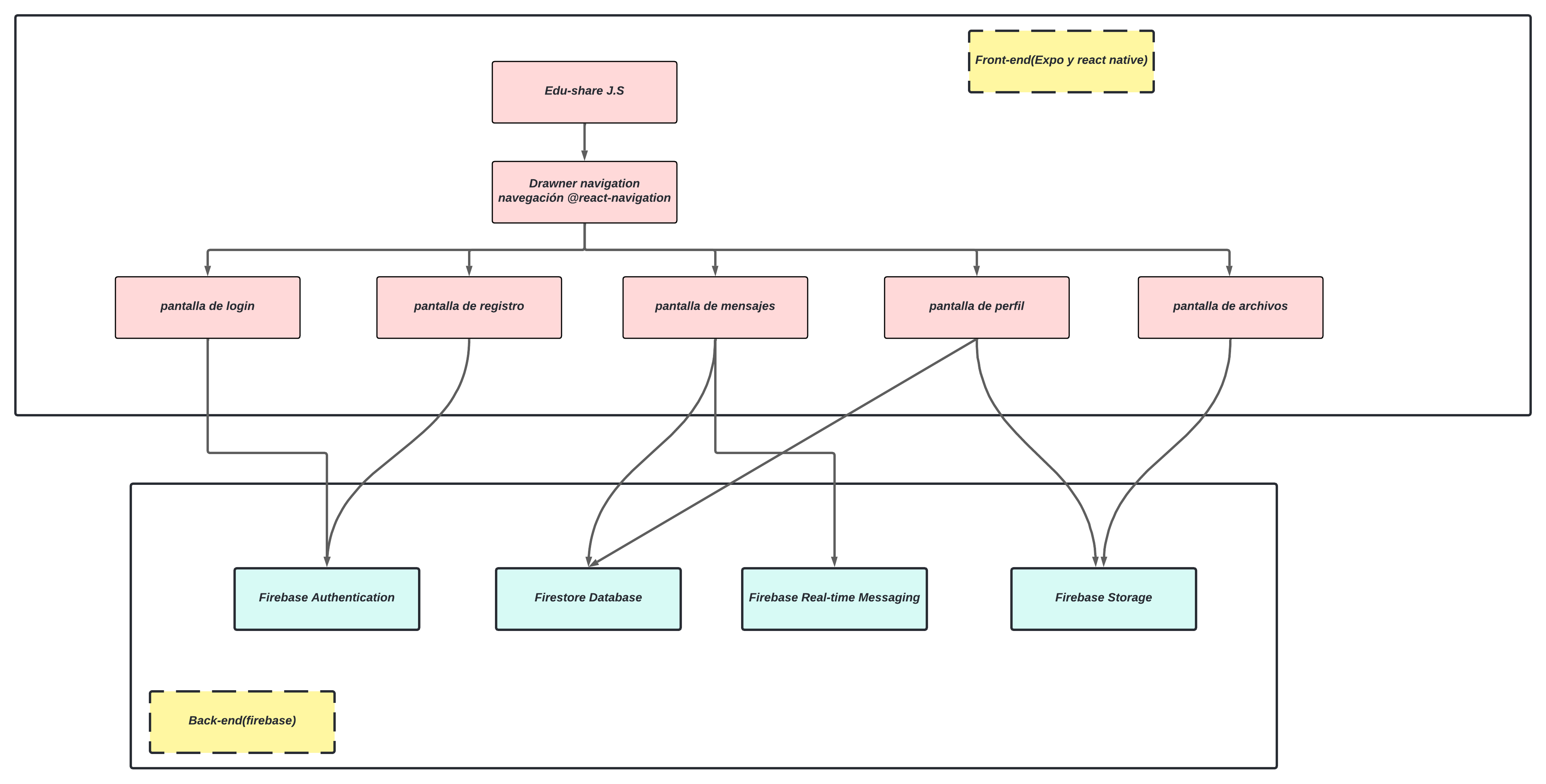
|  |
| --- |
| **jTKgPSIYobxJ0eoAHg0T** |
| {  “createdAt”:” 11 de septiembre de 2024, 12:00:00 a.m. UTC-6”,  “createdBy”:” uhgotn9sNySBfv9QlaxaZzaW”,  “name”:”primer año”,  “participants”: 8 |

# **Arquitectura de Software**

Como para cada proyecto informático se necesita hacer una arquitectura , ya que es el diseño estructurado de los sistemas y aplicaciones para que sean eficientes. En este caso usaremos la arquitectura de:

**Cliente-servidor**

Imagen representativa del modelo con sus elementos claves además de una explicación del mismo.



***Frontend (React Native)***

**Drawer Navigator (@react-navigation).** Este es el sistema de navegación principal de la app. Utiliza el Drawer Navigator de la biblioteca @react-navigation, lo que permite a los usuarios acceder a diferentes secciones de la app deslizando desde el borde de la pantalla o a través de un icono de menú.

***Pantallas***

**Pantalla de Login**. Maneja la autenticación de usuarios.

**Pantalla de Registro.** Permite a nuevos usuarios crear una cuenta.

**Pantalla de Mensajes.** Interfaz para la funcionalidad de chat.

**Pantalla de Perfil.** Muestra y permite editar la información del usuario.

**Pantalla de Archivos.** Nueva adición para manejar la carga y descarga de archivos.

***Backend (Firebase)***

**Firebase Authentication.** Gestiona todo el proceso de autenticación de usuarios, incluyendo registro, inicio de sesión y gestión de sesiones.

**Firestore Database.** Base de datos NoSQL en tiempo real que almacena y sincroniza datos como perfiles de usuario, mensajes y metadatos de archivos.

**Firebase Storage.** Servicio para almacenar y recuperar archivos de usuario, como imágenes de perfil y archivos compartidos.

**Firebase Real-time Messaging**. Facilita la funcionalidad de mensajería en tiempo real para el chat.

***Interacciones***

* Las líneas punteadas representan las interacciones entre las pantallas del frontend y los servicios de Firebase:
* Las pantallas de Login y Registro se comunican con Firebase Authentication.
* La pantalla de Mensajes interactúa con Firestore Database para almacenar/recuperar mensajes y con Firebase Real-time Messaging para actualizaciones en tiempo real.
* La pantalla de Perfil utiliza Firestore Database para datos de usuario y Firebase Storage para imágenes de perfil.
* La nueva pantalla de Archivos se conecta principalmente con Firebase Storage para la gestión de archivos.

***Flujo de Navegación***

El Drawer Navigator actúa como el centro de navegación, permitiendo a los usuarios moverse entre las diferentes pantallas principales de la aplicación.Esto proporciona una experiencia de usuario fluida y coherente en toda la aplicación.

***Arquitectura General***

Esta estructura sigue un modelo cliente-servidor, donde React Native actúa como el cliente (frontend) y Firebase proporciona servicios de backend serverless.La arquitectura permite una separación clara entre la lógica de presentación (frontend) y la lógica de negocio/datos (backend), facilitando el mantenimiento y la escalabilidad.

# **Lógica de negocio/programación**

***Navegación con React Navigation (Drawer Navigation)***

El componente principal (App.js) inicializa la aplicación y define el esquema de navegación, probablemente utilizando un Drawer Navigator (menú lateral) que permite moverse entre las diferentes pantallas como login, registro, mensajes, perfil y archivos.

Cada pantalla se registra en el navegador y se puede acceder a ella mediante la interacción del usuario con el drawer. React Navigation gestiona las transiciones y el historial de navegación.

***Autenticación con Firebase***

El Firebase Authentication se utiliza para gestionar el registro e inicio de sesión de usuarios.

La pantalla de login (Login.js) se conecta a Firebase Authentication para permitir que los usuarios inicien sesión.

La pantalla de registro (Signup.js) se usa para crear nuevas cuentas de usuario. Al registrar o iniciar sesión, se verifica si el usuario existe en la base de datos de Firebase.

Tras el login, se obtiene el token de autenticación para mantener la sesión activa.

***Firestore Database***

Firestore se utiliza para almacenar información adicional del usuario, mensajes y otros datos que forman parte del perfil o las conversaciones.

Cuando un usuario se registra o inicia sesión, su información de perfil se almacena o se recupera de Firestore.

Los mensajes en tiempo real en la pantalla de mensajes (MessagingPage.js) también se almacenan y se leen de Firestore, probablemente con el uso de listeners en tiempo real que actualizan la UI cuando hay nuevos mensajes.

***Firebase Real-time Messaging:***

Usado para el intercambio de mensajes en tiempo real entre los usuarios.

Los mensajes se envían y reciben a través de Firestore y Firebase Real-time Messaging. Los mensajes nuevos activan actualizaciones instantáneas de la interfaz de usuario sin necesidad de refrescar manualmente la página.

***Firebase Storage***

Firebase Storage gestiona el almacenamiento de archivos, como imágenes de perfil o documentos subidos por los usuarios.

En la pantalla de archivos (FilesPage.js), los usuarios pueden cargar y descargar archivos. Los archivos subidos se almacenan en Firebase Storage, y se recuperan utilizando URLs generadas por Firebase para mostrar el contenido o permitir la descarga.

***Manejo de Estados Globales y Contextos***

Se podría estar utilizando React Context API o una librería como Redux para manejar estados globales, como el estado de autenticación del usuario o la información de su perfil.

El estado global asegura que todas las pantallas puedan acceder a datos esenciales como el estado de autenticación del usuario sin necesidad de prop drilling (pasar propiedades de componente en componente).

***Flujo General***

**Inicio de la aplicación.** La aplicación se inicializa en App.js donde se configura la navegación.

**Autenticación.** Si el usuario no está autenticado, se redirige a la pantalla de login. Al iniciar sesión o registrarse, se valida con Firebase.

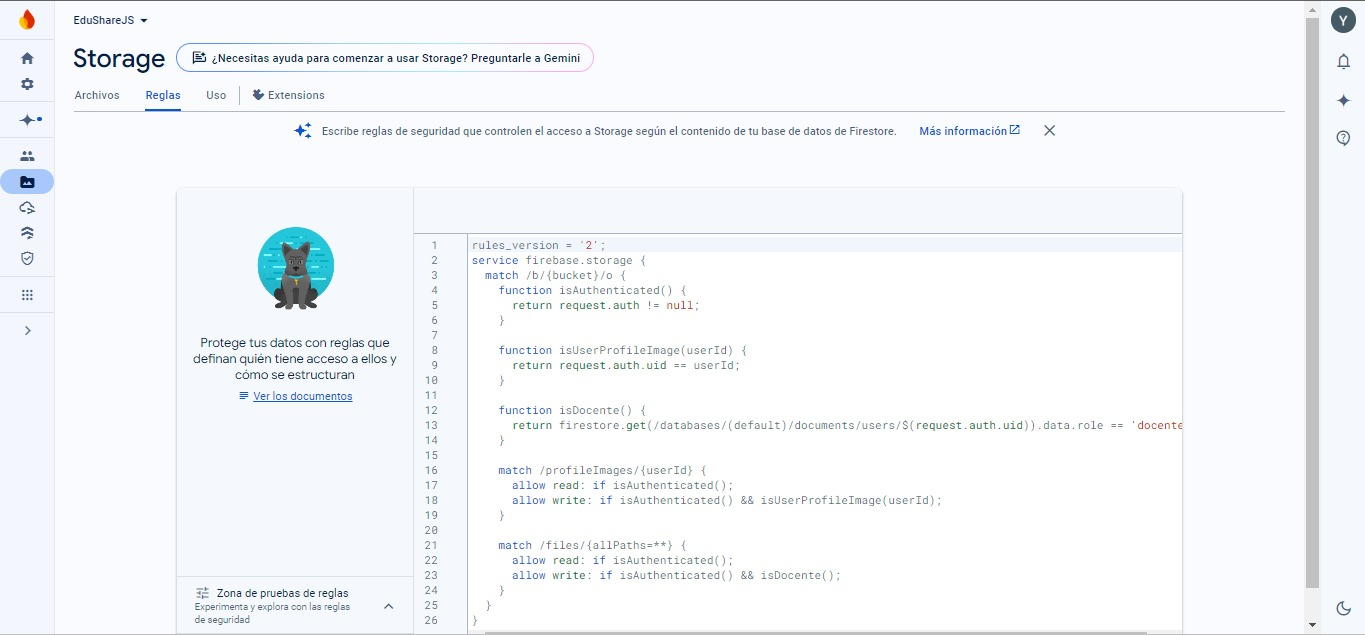
**Navegación.** Una vez autenticado, el usuario puede navegar entre las distintas pantallas (mensajes, perfil, archivos) usando el Drawer Navigator.

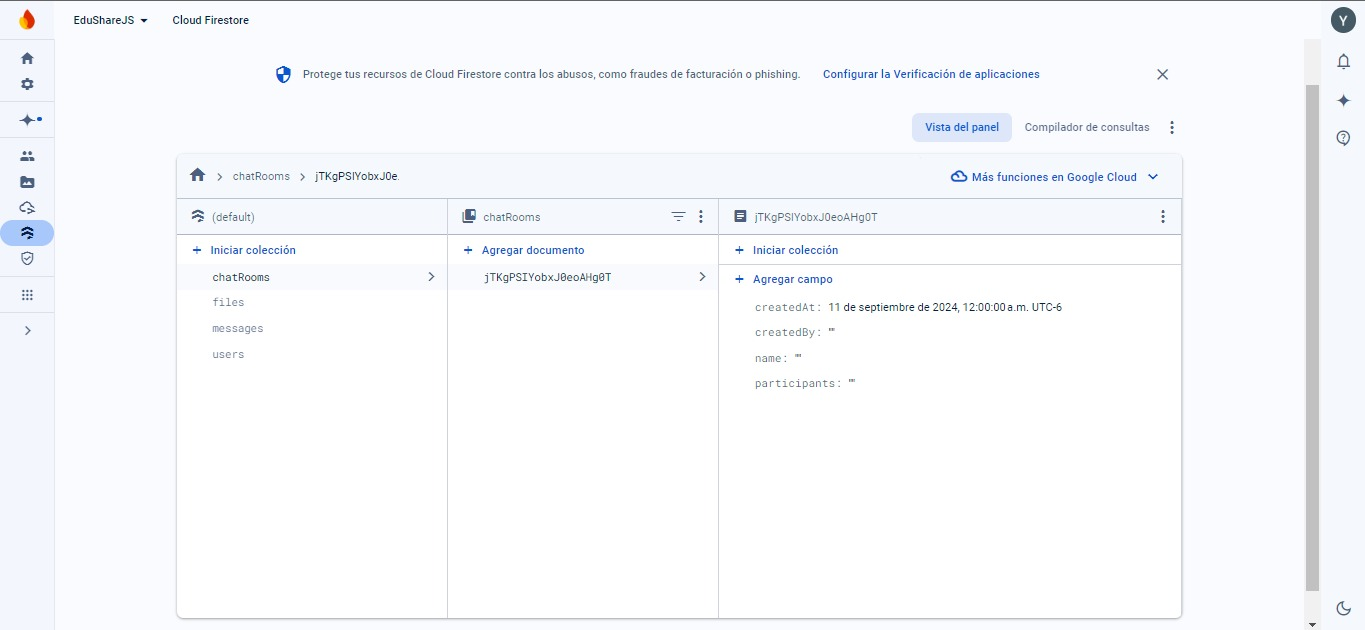
**Mensajes y Archivos.** La aplicación se conecta a Firestore y Firebase Storage para leer/escribir mensajes y subir/descargar archivos.

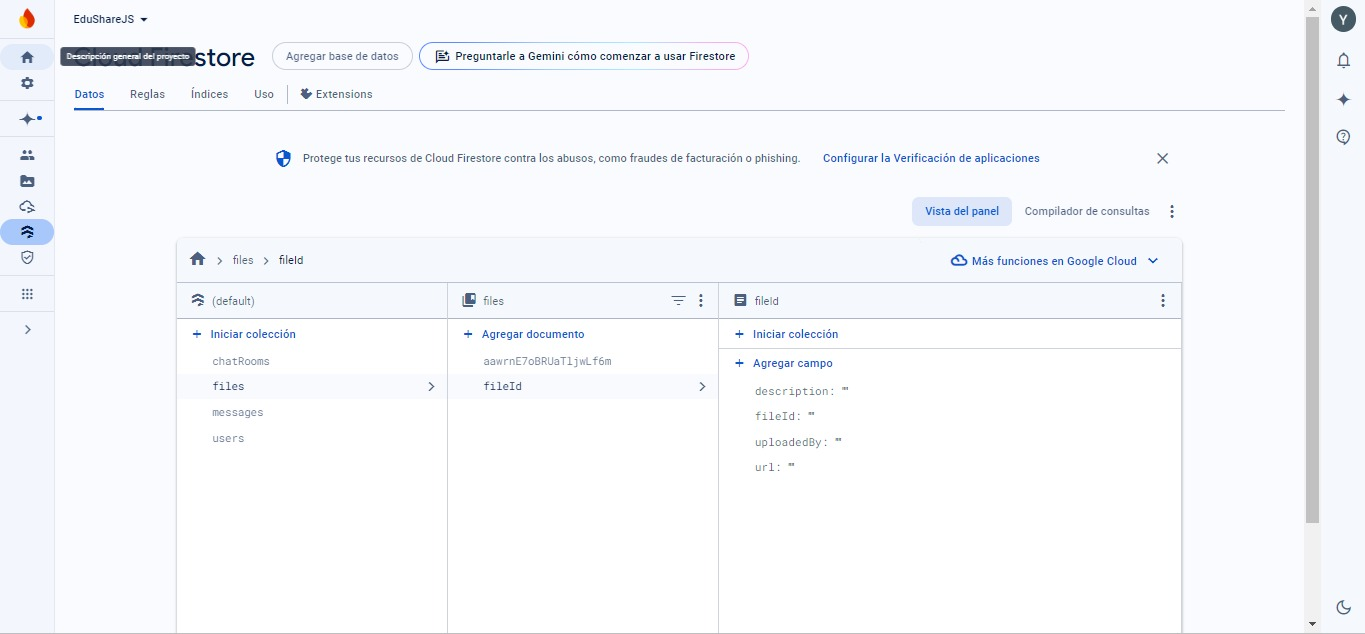
Este enfoque modular y basado en componentes garantiza que cada parte de la aplicación sea mantenible y fácil de escalar.

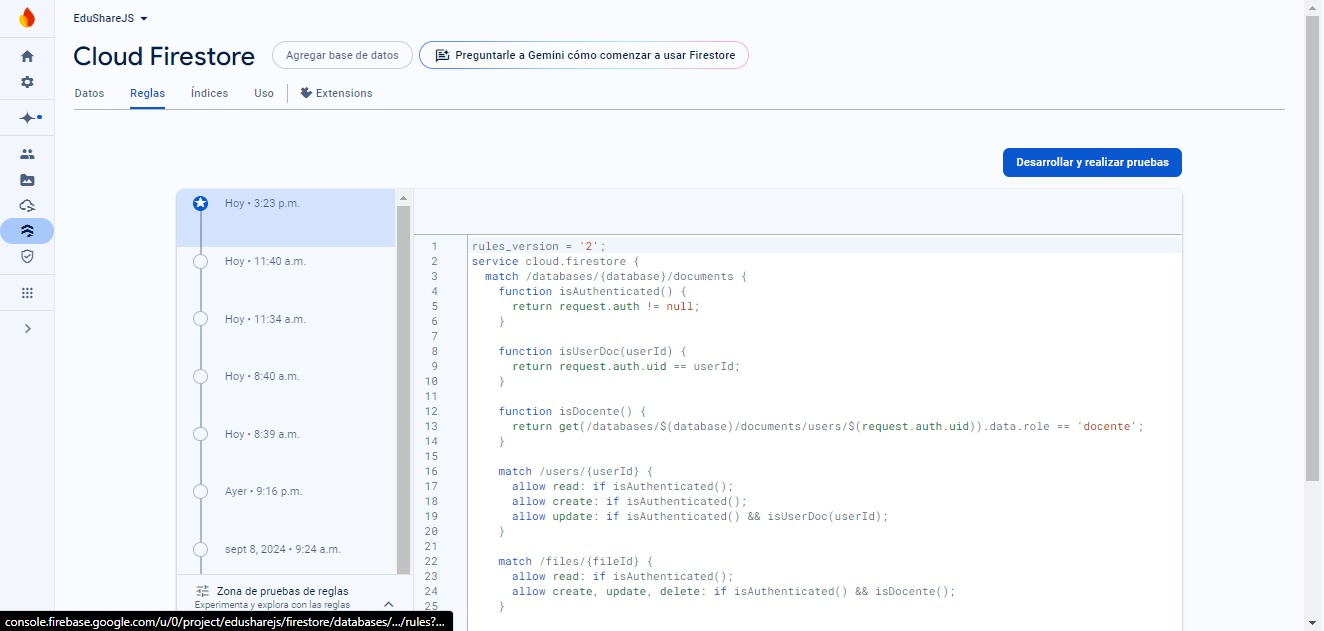
Es una aplicación Multiplataforma que funciona en Android y IOS además que será educativa comunicativa basada en la arquitectura cliente-servidor y operando en la nube. Combina funciones clave como mensajería en tiempo real, autenticación de usuarios, y manejo de archivos, lo que la hace ideal para fomentar la interacción y el intercambio de conocimientos entre estudiantes y profesores, o cualquier comunidad educativa.

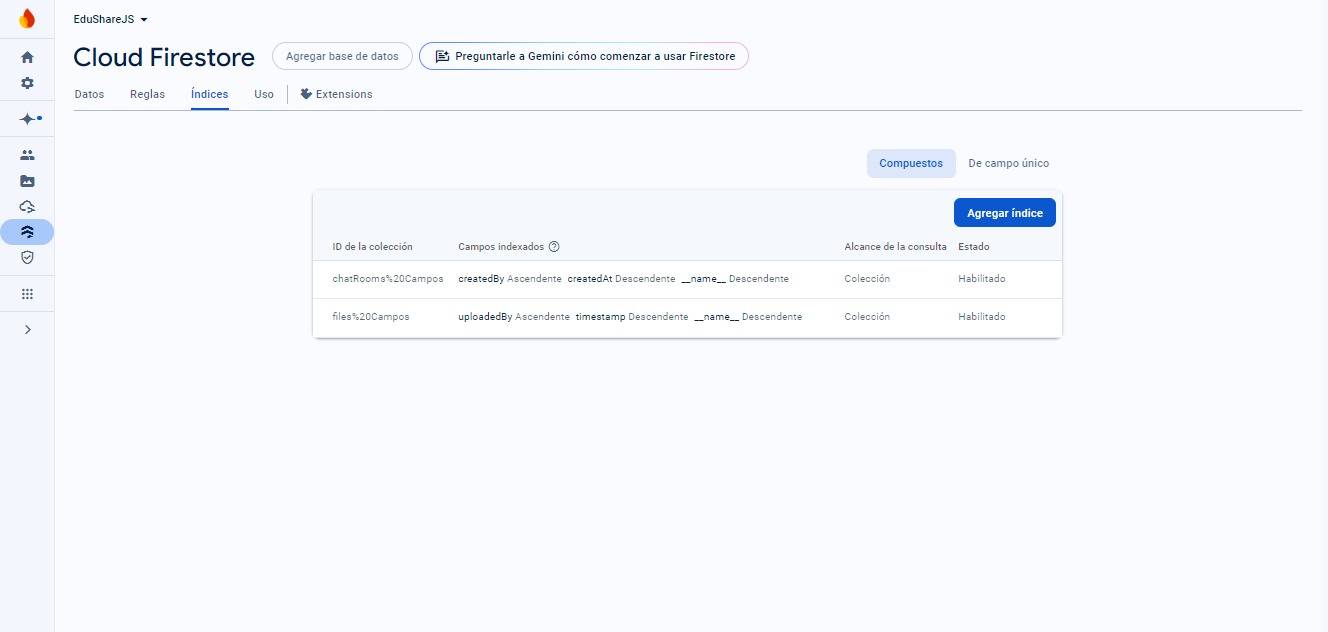
# **BD Fisica evidencias**

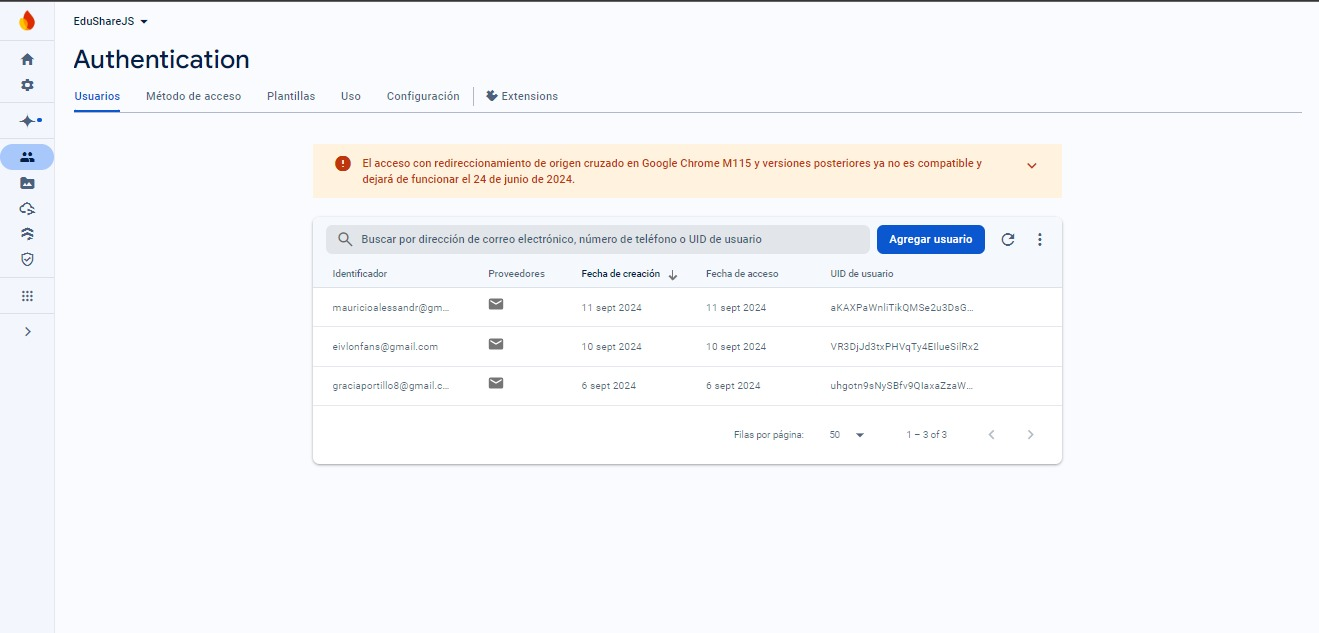
****











# **Contenido original**

El contenido original de nuestro proyecto las imágenes, fotos, videos, texto y formatos originales que usaremos solo son el logo de nuestra app siento hechos con AI.

