

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
FORMULACION DEL PROBLEMA	4
planteamiento del problema	6
Método y técnica de investigación.....	7
Requerimientos de creación y utilización del software.	9
Flutter	9
REACT	24
Estudio de factibilidad.....	34
Modelado de Base de Datos	35
Modelado Clave-Valor	35
Modelo documental.....	36
Arquitectura de Software	39
Lógica de negocio/programación.....	41
BD Fisica evidencias	45

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.

FORMULACION 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 aquellos 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.

Flutter

El proyecto "EduShare.js" busca la forma en que las herramientas educativas interactivas son desarrolladas y compartidas, aprovechando el poder de Flutter, un framework de código abierto de Google que permite crear aplicaciones nativas para múltiples plataformas desde una sola base de código. Este documento aborda los requerimientos técnicos y financieros necesarios para la creación, instalación y mantenimiento del entorno de desarrollo utilizando Flutter 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 creación

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

Flutter es un framework de código abierto desarrollado por Google que permite la creación de aplicaciones nativas para Android, iOS, web y escritorio utilizando un solo código base. Su versatilidad y eficiencia lo han convertido en una opción popular entre los desarrolladores. Sin embargo, para aprovechar al máximo Flutter, 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 Flutter 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 Flutter 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). Flutter 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 Flutter, Android Studio, y otras herramientas necesarias como Visual Studio. Es recomendable tener más espacio disponible para manejar proyectos grandes y dependencias adicionales.

- **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, 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 Flutter, 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 Flutter y Dependencias en Windows

2.1. Descargar Flutter

El primer paso es descargar el SDK de Flutter desde su [sitio oficial](<https://flutter.dev/docs/get-started/install>). Sigue estos pasos para instalar Flutter en Windows:

1. Descargar el SDK:

- Ve al [sitio oficial de Flutter](<https://flutter.dev/docs/get-started/install/windows>) y descarga el archivo ZIP del SDK de Flutter.
- Una vez descargado, descomprime el archivo en una ubicación accesible en tu disco duro, como C:\src\flutter.

2. Configuración de Variables de Entorno:

- Para que Flutter sea accesible desde cualquier ventana de terminal (Command Prompt, PowerShell o Git Bash), es necesario agregar la ruta C:\src\flutter\bin a las variables de entorno del sistema.

Pasos para agregar Flutter al PATH en Windows:

- Abre el Panel de Control y navega a Sistema y Seguridad > Sistema > Configuración avanzada del sistema.
- En la pestaña Opciones avanzadas, haz clic en Variables de entorno....
- En la sección Variables de usuario para <tu-usuario>, selecciona la variable Path y haz clic en Editar.
- Agrega la ruta C:\src\flutter\bin y guarda los cambios.

Este paso es crucial para que puedas ejecutar comandos de Flutter desde cualquier terminal en Windows.

2.2. Instalación de Dependencias Adicionales

Además de Flutter, hay varias dependencias y herramientas que debes instalar para desarrollar aplicaciones multiplataforma en Windows:

- **Git:** Flutter utiliza Git para manejar su código fuente y dependencias. Git se puede instalar desde su [sitio oficial](<https://git-scm.com/>). Durante la instalación de Git, asegúrate de seleccionar la opción que permite agregar Git al PATH del sistema para que sea accesible desde la terminal. Una vez instalado, verifica la instalación ejecutando `git --version` en la terminal.

- **Java Development Kit (JDK):** Java es necesario para la compilación de aplicaciones Android y para la integración con Android Studio. Se recomienda instalar el JDK (Java Development Kit) desde el [sitio oficial de Oracle](<https://www.oracle.com/java/technologies/javase-jdk11-downloads.html>). Durante la instalación, asegúrate de que el instalador configure las variables de entorno adecuadas (JAVA_HOME y PATH). Si no, tendrás que configurarlas manualmente:

Pasos para configurar JAVA_HOME en Windows:

- Abre Configuración avanzada del sistema y haz clic en Variables de entorno....
- En Variables del sistema, haz clic en Nuevo... para crear una nueva variable.
- Ingresa JAVA_HOME como el nombre de la variable y la ruta de instalación del JDK (por ejemplo, C:\Program Files\Java\jdk-<versión>) como el valor de la variable.
- Luego, edita la variable Path en Variables del sistema y agrega %JAVA_HOME%\bin.

- **Android Studio:** 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. Descárgalo desde su [sitio oficial](<https://developer.android.com/studio>).

2.3. Configuración de Android Studio y SDK

Una vez que Android Studio esté instalado, debes configurar el SDK de Android y los emuladores para desarrollar y probar tus aplicaciones en dispositivos Android:

1. Instalación 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. Asegúrate de que la opción Android SDK Command-line Tools esté seleccionada para poder usar las herramientas desde la terminal.

2. Configuración del Emulador:

- Abre AVD Manager en Android Studio (Herramientas > AVD Manager).
- Crea un nuevo dispositivo virtual (AVD) seleccionando un modelo de teléfono Android y una imagen del sistema adecuada (por ejemplo, Pixel 5 con Android 11).
- Este emulador será usado para probar tus aplicaciones Flutter directamente desde la IDE o la línea de comandos.

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 Flutter. Se puede descargar desde el [sitio oficial](<https://code.visualstudio.com/>).

1.Instalación de VS Code:

- Descarga e instala Visual Studio Code. Durante la instalación, selecciona la opción para agregar code al PATH, permitiendo que puedas abrir VS Code desde la terminal.

2. Instalación de Extensiones para Flutter y Dart:

- Abre VS Code y navega al Marketplace de Extensiones (Ctrl+Shift+X).
- Busca e instala las extensiones Flutter y Dart. Estas extensiones proporcionan herramientas adicionales, como el autocompletado de código, la depuración y la integración directa con Flutter.

2.5. Instalación de Visual Studio 2022 Community

Visual Studio 2022 Community es necesario si planeas desarrollar aplicaciones de escritorio (Windows) con Flutter, ya que proporciona las herramientas necesarias para compilar aplicaciones C++.

1. Descarga e Instalación:

- Descarga Visual Studio 2022 Community desde su [sitio oficial](<https://visualstudio.microsoft.com/>).

- Durante la instalación, selecciona el paquete de Desarrollo de aplicaciones de escritorio con C++ (Desktop Development with C++), que incluye compiladores y herramientas necesarias.

2.6. Instalación y Configuración de Firebase

Firebase es una plataforma de Google que proporciona una amplia gama de servicios backend, como bases de datos en tiempo real, autenticación de usuarios, y hosting. Para integrarlo con Flutter en tu proyecto, sigue estos pasos:

1. Instalación de Firebase CLI:

- Utiliza Node.js (que también habrás instalado previamente) para instalar Firebase CLI con el comando: `npm install -g firebase-tools`.

2. Iniciar Sesión en Firebase:

- Autentica tu cuenta de Firebase en la terminal con `firebase login`.

3. Inicializar Firebase en tu Proyecto:

- Navega a la carpeta raíz de tu proyecto Flutter y ejecuta `firebase init`. Esto te permitirá configurar Firebase en tu proyecto y seleccionar los servicios que deseas utilizar (como Firestore, Authentication, etc.).

2.7. 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:

- Flutter: C:\src\flutter\bin
- Git: C:\Program Files\Git\bin
- Java: C:\Program Files\Java\jdk-<versión>\bin
- Android SDK: C:\Users\<tu-usuario>\AppData\Local\Android\Sdk\platform-tools
- Node.js: C:\Program Files\nodejs\

Para verificar que todas las herramientas estén correctamente configuradas, ejecuta flutter doctor en la terminal. Este comando revisará la configuración del entorno de Flutter 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 Flutter

EduShare.js es un proyecto innovador en Flutter 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.

1. 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.

2. 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.

3. 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 Flutter

EduShare.js es una plataforma desarrollada en Flutter 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:

1. 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.

2. 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.

3. Memoria RAM

El desarrollo en Flutter, 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.

4. Firebase

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 Flutter y de la disponibilidad de recursos tecnológicos y financieros. Desde la instalación de Flutter 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.

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)

No	RECURSOS TECNICOS O HERRAMIENTAS	COSTO O VALOR UNITARIO	CANTIDAD Y UNIDAD	SUBTOTAL(VALOR)	SUBTOTAL(COSTO REAL)
1	Laptop Dell Latitude 3120	\$210.00	6	\$1,260.00	\$0.00
2	Lenovo thinkpad w541	\$449.00	1	\$449.00	\$0.00
3	Trello (Software de organización)	\$0.00	1	\$0.00	\$0.00
4	Flutter(Software de desarrollo)	\$0.00	1	\$0.00	\$0.00
5	React Native(Software de desarrollo)	\$0.00	1	\$0.00	\$0.00
6	Firebase (Alojamiento web)	\$0.00	1	\$0.00	\$0.00
7	Visual Studio Code(Editor de código)	\$0.00	1	\$0.00	\$0.00
8	Android Studio(Emulador)	\$0.00	1	\$0.00	\$0.00
TOTAL	INVERSION TOTAL DE EL PROYECTO EN HERRAMIENTAS (VALOR/COSTO REAL)			\$1,709.00	\$0.00
No	SERVICIOS	COSTOS	CANTIDAD(MESES)	SUBTOTAL(VALOR)	SUBTOTAL(COSTO REAL)
9	Internet	\$25.00	4	\$100.00	\$100.00
10	Energia electrica	\$30.00	4	\$120.00	\$120.00
TOTAL	INVERSION TOTAL DE EL PROYECTO EN SERVICIOS (VALOR/COSTO REAL)			\$220.00	\$220.00
No	SERVICIOS	COSTOS	CANTIDAD	SUBTOTAL(VALOR)	SUBTOTAL(COSTO REAL)
11	Analista de Sistemas	\$1,000.00	1	\$1,000.00	\$0.00
12	Desarrollador Back End	\$1,200.00	1	\$1,200.00	\$0.00
13	Desarrollador Front End	\$1,000.00	1	\$1,000.00	\$0.00
14	Diseñador de Interfaces de Usuario (UI)	\$900.00	1	\$900.00	\$0.00
15	Desarrollador de Bases de Datos	\$1,300.00	1	\$1,300.00	\$0.00
16	Full Stack	\$1,500.00	1	\$1,500.00	\$0.00
17	Diseñador de Bases de Datos	\$1,300.00	1	\$1,300.00	\$0.00
TOTAL	INVERSION TOTAL DE EL PROYECTO EN RECURSOS OPERATIVOS (VALOR/COSTO REAL)			\$8,200.00	\$0.00
TOTAL	INVERSION TOTAL DE EL PROYECTO EN RECURSOS OPERATIVOS EN 4 MESES(VALOR/COSTO REAL)			\$32,800.00	\$0.00
TOTAL	INVERSION TOTAL DE EL PROYECTO (VALOR/COSTO REAL)			\$34,729.00	\$220.00

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 authentication , 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

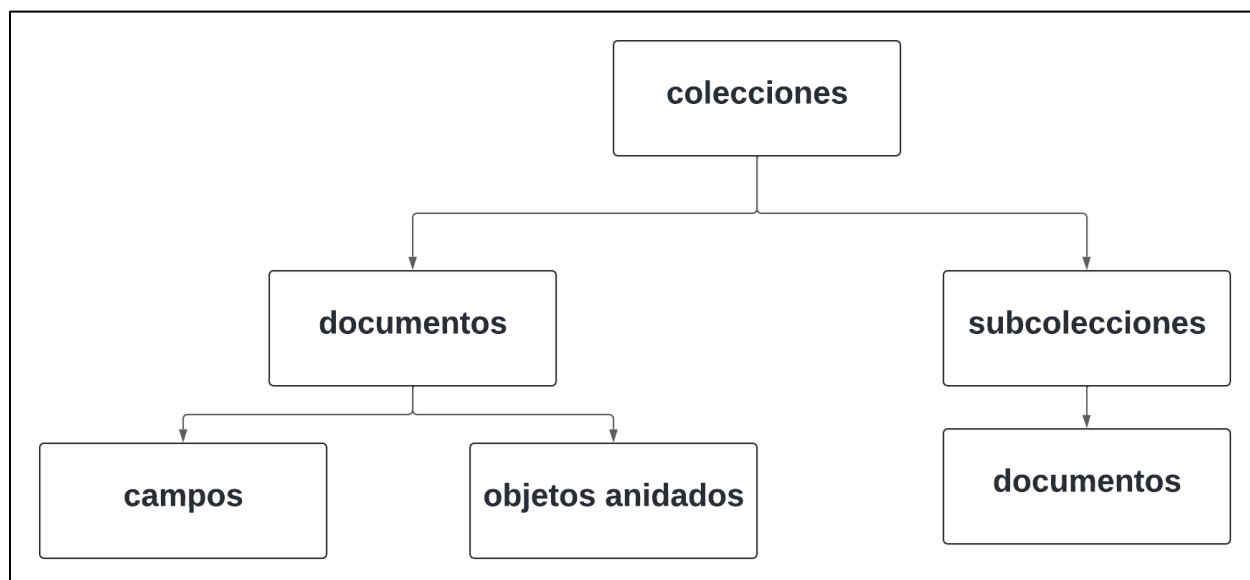
Es como un diccionario de datos gigante donde cada clave tiene un identificador único y cada clave esta asociada a un valor que puede ser cualquier dato . Esto se usa en firebase authentication.

Representación de los datos.

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

Modelo documental

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



Colecciones. Son los contenedores de los documentos.

Documentos. Son 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

ws6nk2FVD0bQhnTBjüeei

```
{
  "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

jTKgPSIYobxJOeoAHg0T

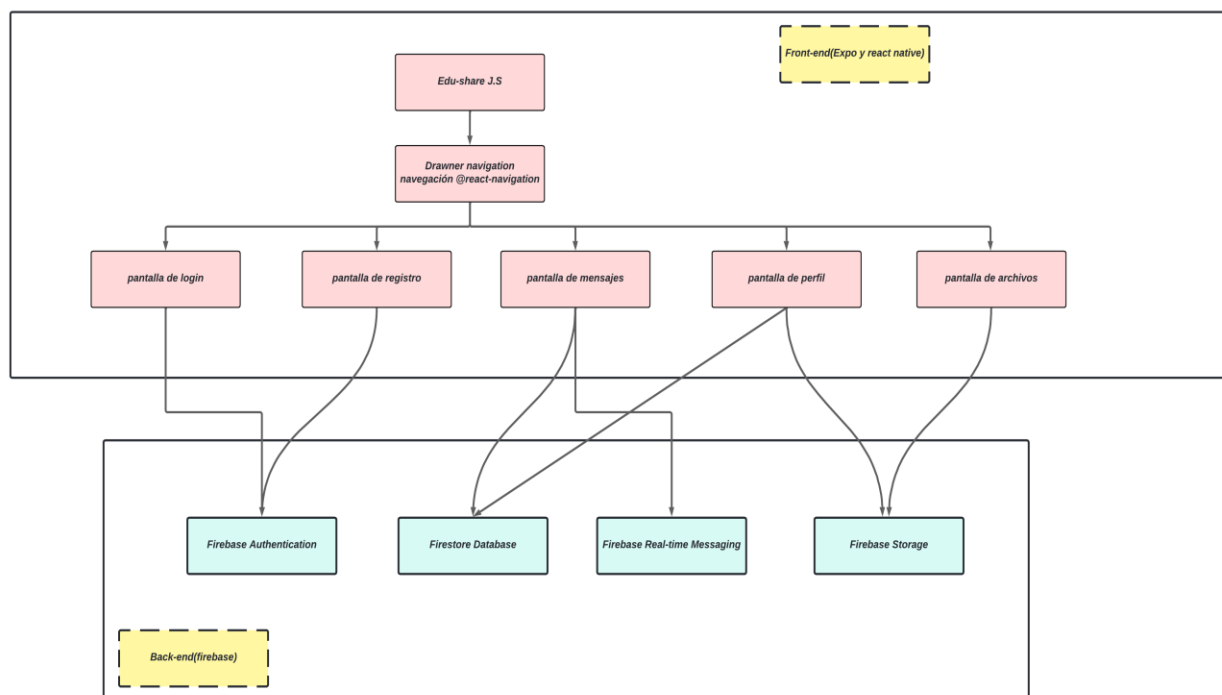
```
{
  "createdAt": "11 de septiembre de 2024, 12:00:00 a.m. UTC-6",
  "createdBy": "uhgotn9sNySBfv9QlaxaZzaW",
  "name": "primer año",
  "participants": 8
}
```

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:

Ciente-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.

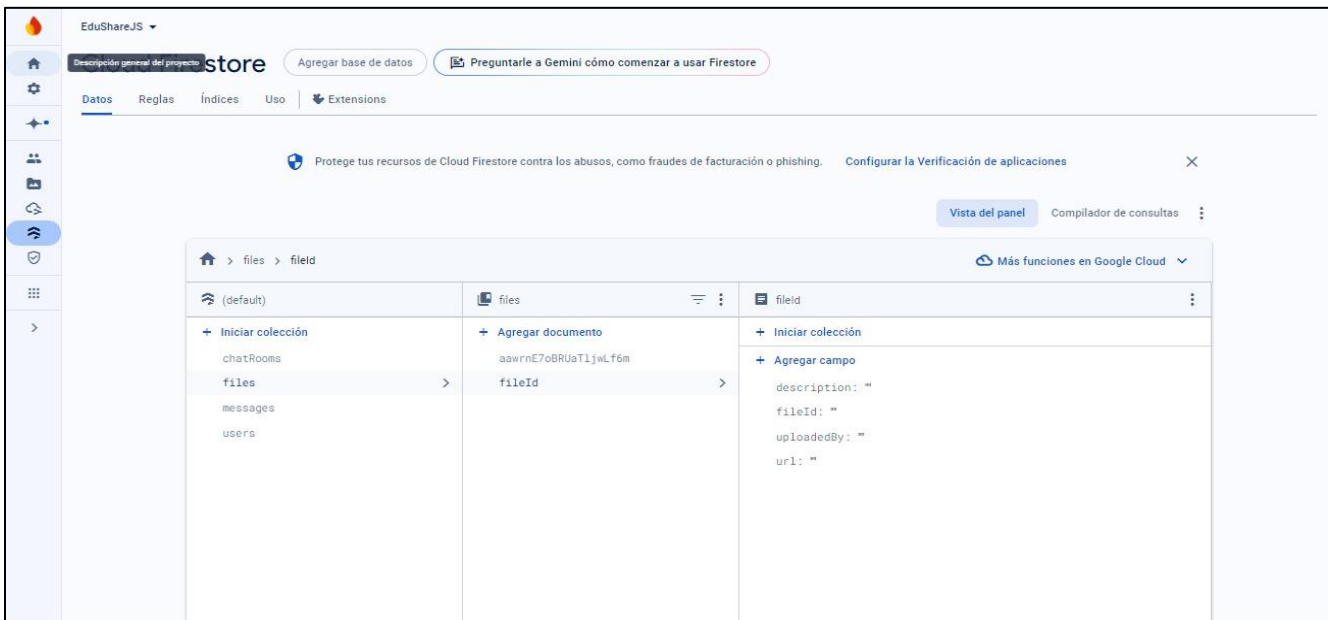
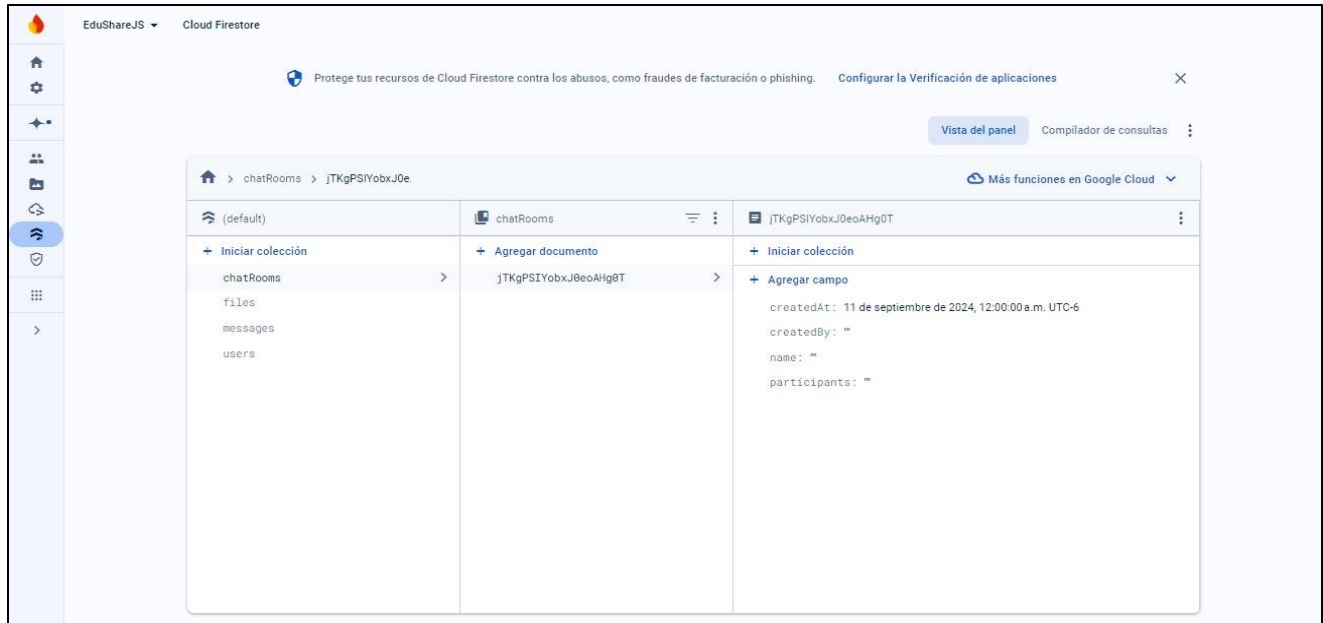
Es una aplicación educativa comunicativa basada en la arquitectura cliente-servidor y operando en la nube. Combina funciones clave como mensajería en tiempo real, autenticación de usuarios, y manejo de archivos, lo que la hace ideal para fomentar la interacción y el intercambio de conocimientos entre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

BD Fisica evidencias

The screenshot displays the Firebase Storage console interface. At the top, the 'Storage' tab is selected, with a sub-tab 'Reglas' (Rules) active. A sidebar on the left contains navigation icons. The main content area features a 'Zona de pruebas de reglas' (Rules test zone) on the left, which includes a warning icon and text: 'Protege tus datos con reglas que definan quién tiene acceso a ellos y cómo se estructuran' (Protect your data with rules that define who has access to them and how they are structured), followed by a link 'Ver los documentos' (View the documents). To the right of this is a code editor showing the following security rules:

```
1 rules_version = '2';
2 service firebase.storage {
3   match /b/{bucket}/o {
4     function isAuthenticated() {
5       return request.auth != null;
6     }
7
8     function isUserProfileImage(userId) {
9       return request.auth.uid == userId;
10    }
11
12    function isDocente() {
13      return firestore.get(/databases/(default)/documents/users/{request.auth.uid}).data.role == 'docente';
14    }
15
16    match /profileImages/{userId} {
17      allow read: if isAuthenticated();
18      allow write: if isAuthenticated() && isUserProfileImage(userId);
19    }
20
21    match /files/{allPaths=**} {
22      allow read: if isAuthenticated();
23      allow write: if isAuthenticated() && isDocente();
24    }
25  }
26 }
```

At the bottom of the code editor, there is a 'Zona de pruebas de reglas' (Rules test zone) with the text 'Experimenta y explora con las reglas de seguridad' (Experiment and explore with the security rules).



Cloud Firestore | Agregar base de datos | Preguntarle a Gemini cómo comenzar a usar Firestore

Datos | **Reglas** | Índices | Uso | Extensions

Desarrollar y realizar pruebas

Timeline:

- Hoy • 3:23 p.m.
- Hoy • 11:40 a.m.
- Hoy • 11:34 a.m.
- Hoy • 8:40 a.m.
- Hoy • 8:39 a.m.
- Ayer • 9:16 p.m.
- sept 8, 2024 • 9:24 a.m.

Reglas:

```
1 rules_version = '2';
2 service cloud.firestore {
3   match /databases/{database}/documents {
4     function isAuthenticated() {
5       return request.auth != null;
6     }
7
8     function isUserDoc(userId) {
9       return request.auth.uid == userId;
10    }
11
12    function isDocente() {
13      return get(/databases/{database}/documents/users/{request.auth.uid}).data.role == 'docente';
14    }
15
16    match /users/{userId} {
17      allow read: if isAuthenticated();
18      allow create: if isAuthenticated();
19      allow update: if isAuthenticated() && isUserDoc(userId);
20    }
21
22    match /files/{fileId} {
23      allow read: if isAuthenticated();
24      allow create, update, delete: if isAuthenticated() && isDocente();
25    }
26  }
```


console.firebase.google.com/u/0/project/edusharejs/firestore/databases/.../rules?...

Cloud Firestore | Agregar base de datos | Preguntarle a Gemini cómo comenzar a usar Firestore











Datos | Reglas | **Índices** | Uso | Extensions

Compuestos | De campo único | **Agregar índice**

ID de la colección	Campos indexados	Alcance de la consulta	Estado
chatRooms%20Campos	createdBy Ascendente createdAt Descendente __name__ Descendente	Colección	Habilitado
files%20Campos	uploadedBy Ascendente timestamp Descendente __name__ Descendente	Colección	Habilitado



EduShareJS



Authentication

Usuarios


Método de acceso


Plantillas

Uso


Configuración


Extensions




 El acceso con redireccionamiento de origen cruzado en Google Chrome M115 y versiones posteriores ya no es compatible y dejará de funcionar el 24 de junio de 2024.

 Buscar por dirección de correo electrónico, número de teléfono o UID de usuario

Agregar usuario





Identificador	Proveedores	Fecha de creación ↓	Fecha de acceso	UID de usuario
mauricioaleesandr@gmail...		11 sept 2024	11 sept 2024	aKAXPaWnItikQMSe2u30sG...
eivlonfans@gmail.com		10 sept 2024	10 sept 2024	VR3DjJd3txPHVqTy4EllueSlRv2
graciaportillo8@gmail.c...		6 sept 2024	6 sept 2024	uhgotn9aNySBfv9QlaxaZzeW...

Filas por página:

50

1 - 3 of 3

