WeConnect

James
Hey hon, where are you?
...my stomach lol
hahaha you silly
I feel like making tacos! What do you say?
Yay! Could you get me some raspberry ice cream too? Pretty please?

Let Us Chat

Overview

WeConnect is a complete cross-platform online chatting application written in the Flutter framework and the Dart programming language. This application provides a seamless user experience while offering one-to-one chatting, text messaging, photo uploading, and an integrated AI assistant powered by the Gemini API. Firebase powers this for all backend services related to authentication, real-time databases, cloud storage, and notification services.

Architecture

Client-Server Architecture

- **Frontend**: Developed using Flutter, it provides a responsive and smooth user interface that works across multiple platforms (Android, iOS, Web, etc.).
- **Backend**: Firebase serves as the backend, handling data storage, real-time database operations, authentication, and push notifications.
- **AI Assistant**: Integrated using the Gemini API, which processes user input and provides AI-driven responses.

Data Flow

1. **User Registration/Authentication**: User information is kept securely in Firebase Authentication. After successful login, the user will be routed to the main interface for chatting.
2. **Chatting (Text Messaging and Photo Upload)**: Messages and photos are kept in Firebase Firestore for real-time updates and Firebase Storage for media files. Each Chat has a unique identifier to maintain message integrity.
3. **Notification**: Push notifications are made using Firebase Cloud Messaging and Flutter Notification Channel package for high-priority messages.
4. **AI Assistant Integration**: Gemini API handles the queries of the user, and the responses are inducted back into the chat interface.

Technology Stack

- **Flutter**: A UI toolkit for building natively compiled applications for mobile, web, and desktop from a single codebase.
- **Dart**: The programming language used with Flutter.
- **Firebase**:
    - **Authentication**: Manages user identities.
    - **Firestore**: A NoSQL document database to store and sync data for client- and server-side development.
    - **Storage**: For storing media files like photos.
    - **Cloud Messaging**: Handles push notifications.
- **Gemini API**: Provides AI-driven responses for the chat assistant.

- **Flutter Notification Channel**: Manages and registers notification channels for better user experience.

Dependencies and Libraries

Firebase Core

- Initializes Firebase in the Flutter app.
- Used because Firebase provides a comprehensive suite of backend services, crucial for chat applications.

Flutter Notification Channel

- Manages notification channels for Android.
- Ensures notification categorization into their various priorities or levels of importance and visual representations thereof.

System Chrome

- Used to implement the system UI, including full-screen mode, setting the orientation to portrait locked.
- Ensures a consistent user experience between devices.

Firebase Options

- Holds platform-specific Firebase configuration settings.
- The following depends on the platform on which Firebase needs to be initialized correctly: Android, iOS, Web, etc.

Setup and Run Instructions

Prerequisites

- **Flutter SDK**: Install the latest version of the Flutter SDK.
- **Dart SDK**: Comes bundled with the Flutter SDK.
- **Firebase CLI**: Required to initialize and manage Firebase projects.
- **Android Studio/ Xcode/ VS Code**: This is for building and testing Android/iOS devices.

Clone the Repository

```
git clone https://github.com/satyam1917/WeConnect
cd weconnect
```

Firebase Configuration

- Ensure that Firebase is set up for each platform by configuring the google-services.json (Android) and GoogleService-Info.plist (iOS) files.
- These files can be obtained from the Firebase Console under your project settings.

Install Dependencies

```
flutter pub get
```

Run the Application

For Android:
```
flutter run
```
For iOS:
```
flutter run
```
For Web:
```
flutter run -d chrome
```

Troubleshooting

- Ensure that all Firebase services are correctly set up.
- Verify internet connectivity.
- Check for platform-specific issues in the Flutter documentation.

Code Analysis and Functionality

AI Screen (ai_screen.dart)

- This is the interface through which a user can interact with an AI assistant using WeConnect.
- It maintains user input through _textC, scroll management through _scrollC, and conversations through _list.
- The _askQuestion function processes the user's input, refreshes the UI, and fetches back the AI responses using the Gemini API.
- UI elements include a scaffold with an AppBar, a floating action button for queries, and a message ListView.

Chat Screen (chat_screen.dart)

- Allows one-on-one communication between users, supporting messages with text, emoji, and image attachments.
- State-controlled by _list, _textController, _showEmoji, and _isUploading.
- Firebase makes it possible to update data in real-time. So, when a message is sent, in the very second, it will appear on the chat screen.
- UI-APPBAR: A UI AppBar containing user information.
- ListView to render a scrollable list of messages and TextField for inputting a message.

Code Dependencies

- Uses Google Generative AI, Emoji Picker, and Image Picker to enhance the chat experience.

File Breakdown

- **Home Screen (home_screen.dart)**: Displays all available contacts and manages user interactions like searching, adding contacts, and navigating to profiles.
- **Profile Screen (profile_screen.dart)**: Allows users to view and customize their profiles, including updating their profile picture and logging out.
- **Splash Screen (splash_screen.dart)**: Manages the app's initial loading, checking user authentication before navigating to the home or login screen.
- **View Profile Screen (view_profile_screen.dart)**: Displays detailed information about other users, including their profile and join date.

Backend Integration

- **APIs (apis.dart)**: Handles Firebase functionalities like authentication, Firestore database interactions, Firebase Storage for images, and Firebase Cloud Messaging for push notifications.
- **Notification Access (notification_access.dart)**: Manages Firebase admin tokens required for sending push notifications using Google APIs.

Project Configuration

- **pubspec.yaml**: This is the manifest file for this project listing project dependencies and metadata. It includes primary essentials like Firebase, Google Sign-In, Image Picker, etc., that would ensure the very core functionalities of the app are covered.

Atomic Design

WeConnect utilizes Atomic Design in the development of its interface, which is highly scalable and maintainable by breaking down the interface into its most minimalistic components. The design system follows a top-down hierarchy, with atoms being the tiniest building blocks comprising buttons and input fields. These atoms combine to create molecules-for example, a search bar, which then further aggregates into organisms, such as the chat input section. These become then organized into templates, which are forms of defining screen structure that will, in turn, compose the pages-complete UI screens. This makes sure that consistency, reusability, and clarity are enforced throughout the application's interface.

REST APIs

REST APIs act as an integral part of WeConnect, as they provide seamless connectivity from the frontend to the backend. In a RESTful architecture, applications interact with Firebase services regarding users' authentication, retrieval, and storing of chat messages, image upload, and sending notifications. This means each call to the RESTful API follows the standards of HTTP: GET, POST, PUT, and DELETE. This is different but ensures safety and efficacy for data interchange between the client and server. By doing so, this will leave room for flexibility and extension in the application. So, it can be updated in real-time for seamless user interaction.

Future Enhancements

API Key Management: The Gemini API key is hardcoded, which is not secure for production. Future versions will store keys securely, possibly using environment variables.

Data Privacy: Messages and images are stored in Firebase, requiring strict access control to protect user data. Future versions will implement OAuth2 for better authentication.

User Experience: To add features like typing indicators, group chats, and enhanced UI elements.

Scalability: To optimize Firebase usage, and consider server-side processing for handling more data and users.

AI Enhancement: To explore additional AI services for more personalized responses.

Conclusion

WeConnect is a cutting-edge, scalable chat application using the best of modern technologies for a very immersive and seamless user experience. Firebase powering the backend from real-time data synchronization down to secure authentication and cloud storage vows robust performance and reliability. Putting Flutter into action for cross-platform development hastens its deployment through Android, iOS, and the web with consistency in responsiveness. The code for WeConnect is extendable and flexible for further improvements and big-scale user engagement. With AI-driven responses, real-time messaging, and picture sharing securely, it may prove quite flexible in solving modern communication needs.

Proof of Concept

Apk:

https://drive.google.com/file/d/1SXojXYymxgRIfH7aQkiPcvr9Ta0nlOwJ/view?usp=sharing

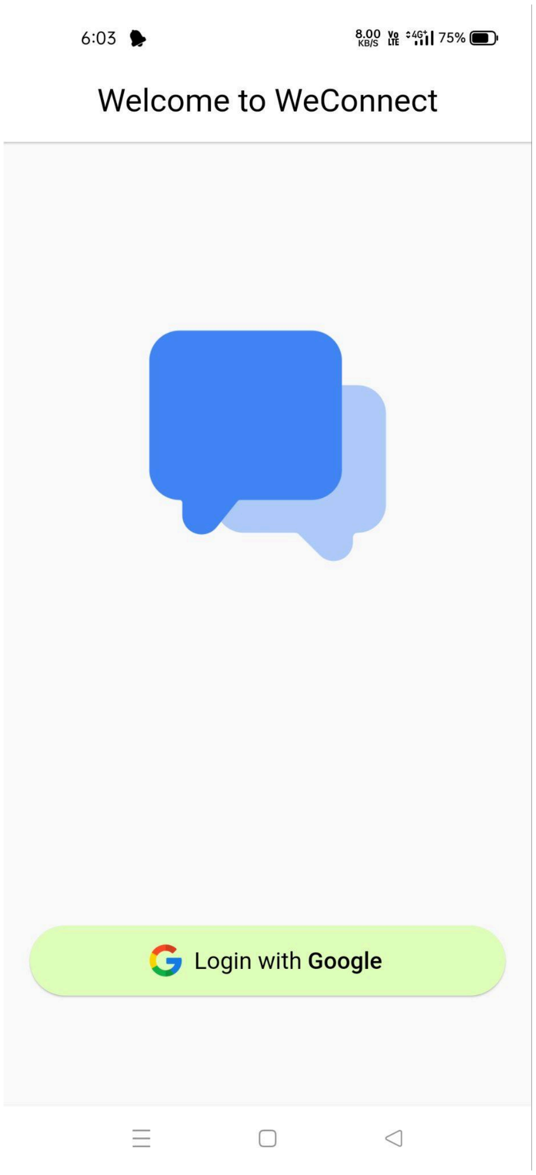
Demo Video Link:

https://github.com/satyam1917/WeConnect/blob/main/Demo_Video.mp4

Snapshots:

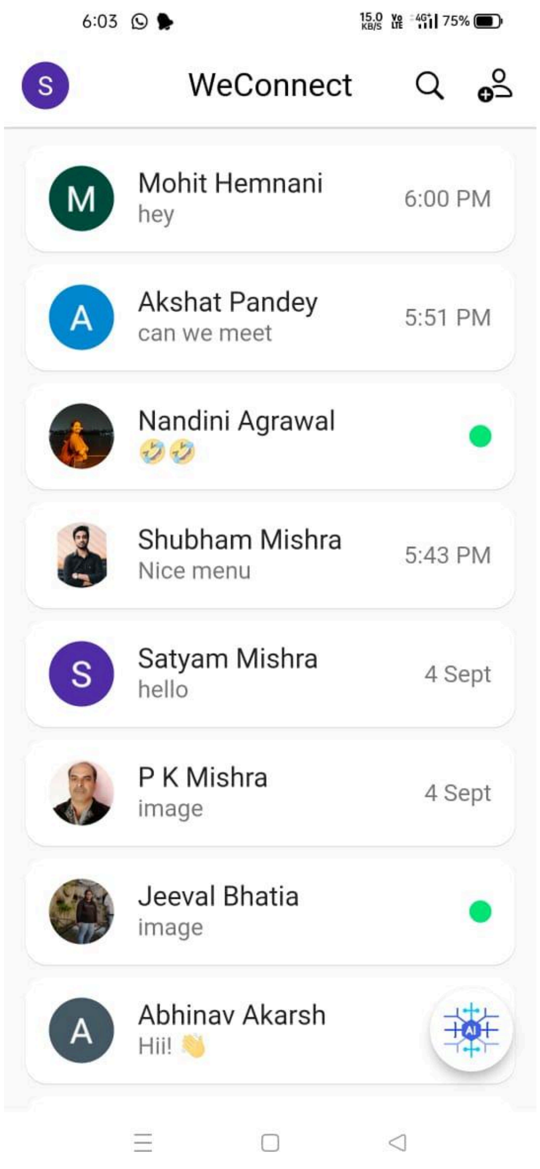
https://github.com/satyam1917/WeConnect/tree/main/SnapShots

Please find below some of the snapshots of important screens of the app.



Welcome to WeConnect

Login with **Google**



WeConnect

Mohit Hemnani
hey
6:00 PM

Akshat Pandey
can we meet
5:51 PM

Nandini Agrawal
🤣🤣
●

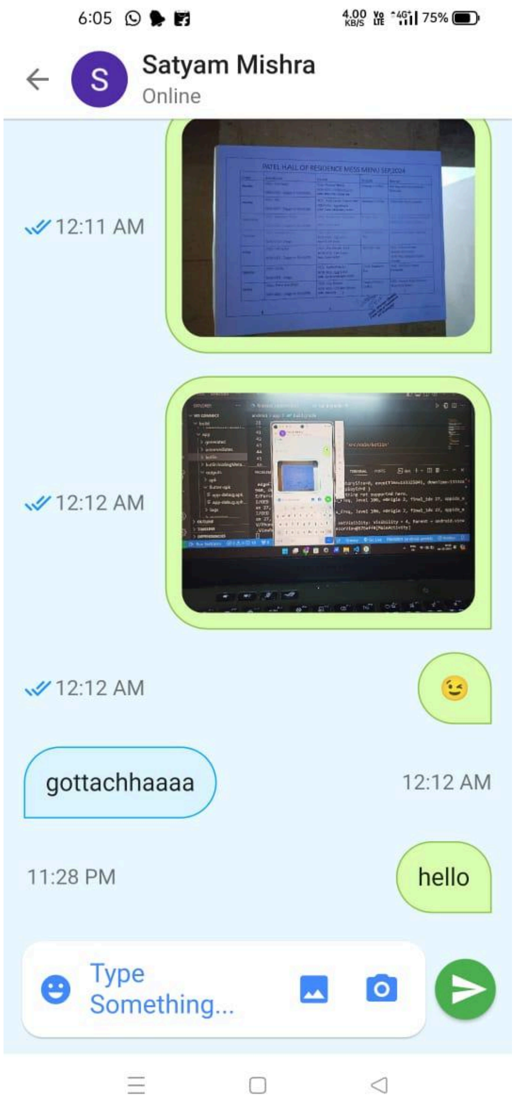
Shubham Mishra
Nice menu
5:43 PM

Satyam Mishra
hello
4 Sept

P K Mishra
image
4 Sept

Jeeval Bhatia
image
●

Abhinav Akarsh
Hii! 👋



6:05 4.00 KB/S 75%

Satyam Mishra
Online

12:11 AM

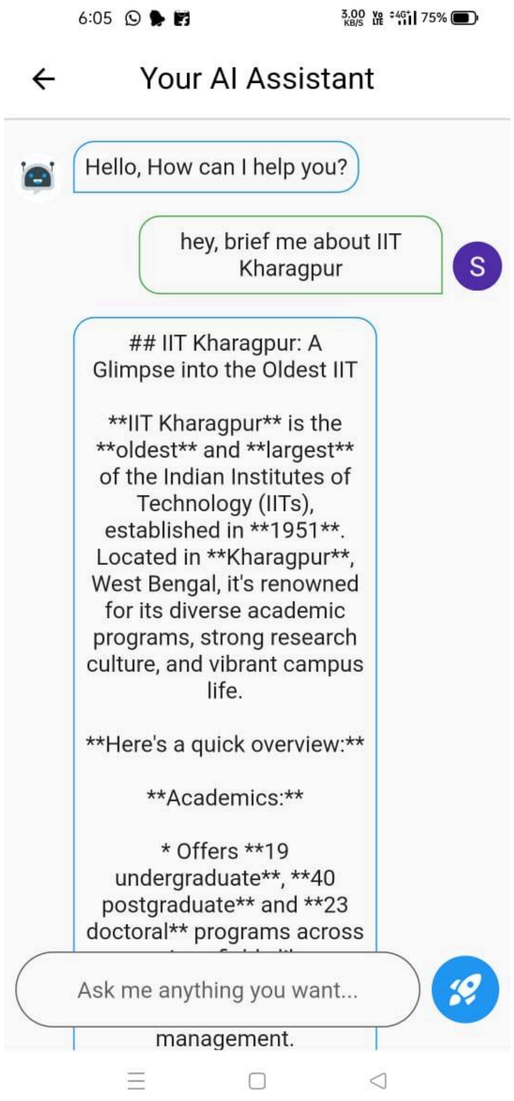
12:12 AM

12:12 AM 😉

gottachhaaaa 12:12 AM

11:28 PM hello

Type Something...



6:05 3.00 KB/S 75%

Your AI Assistant

Hello, How can I help you?

hey, brief me about IIT Kharagpur

IIT Kharagpur: A Glimpse into the Oldest IIT

IIT Kharagpur is the **oldest** and **largest** of the Indian Institutes of Technology (IITs), established in **1951**. Located in **Kharagpur**, West Bengal, it's renowned for its diverse academic programs, strong research culture, and vibrant campus life.

Here's a quick overview:

Academics:

* Offers **19 undergraduate**, **40 postgraduate** and **23 doctoral** programs across

Ask me anything you want...

management.