ChatConnect Using Android Studio and Firebase

1. Project Objectives:

- **Purpose**:

  ChatConnect is designed as a sample Android chat application built with the Compose UI toolkit to demonstrate a basic, real-time chat experience. It allows users to create accounts, set up profiles, and engage in one-on-one conversations with other account holders.

- **Goals**:

  - Showcase the capabilities of Compose UI for creating responsive and modern UIs.
  - Integrate Firebase to handle user authentication, real-time data, and storage for user profiles.
  - Provide a functional example of a chat interface with smooth input handling, navigation, and state management.
  - Implement mobile number authentication to ensure secure and straightforward user logins.

2. Functionalities:

a) User Authentication

- **Mobile Number Login**:

  - ChatConnect uses Firebase Authentication for user logins, allowing users to authenticate via their mobile numbers.
  - When users enter their mobile number, an OTP (One-Time Password) is sent for verification.
  - Once verified, the user is granted access to their account or prompted to set up a new profile if it's their first time logging in.

b) Profile Setup

- **Profile Picture and Display Name**:

  - Users can upload a profile picture using an image picker component in the app.
  - Firebase Storage is used to store the profile images securely, while user metadata (such as display name) is saved in Firestore Database.
  - The Compose UI handles the image display, enabling a seamless user experience as they select or update their profile picture.

c) Chat Functionality

- **One-on-One Chat**:
  - Users can initiate and participate in one-on-one text conversations with any other account holders in the app.
  - Messages are stored and retrieved in real-time using Firebase Firestore, enabling instant communication.
  - The chat interface leverages Compose UI for managing message lists, with messages aligned based on sender and receiver status.
- **Input Handling**:
  - Compose's state management system is utilized to manage text inputs and handle messages in real-time.
  - The interface is designed to clear the input field upon sending a message, ensuring a fluid chat experience.
- **Navigation between Screens**:
  - Navigation within ChatConnect is handled using Compose's NavHost and NavController to move between login, profile setup, and chat screens.
  - Back stack management ensures that users can return to previous screens easily and intuitively.

3. Additional Requirements:

a) Firebase Configuration and Project Setup

- **Firebase Setup**:
  1. Create a Firebase project at the Firebase Console.
  2. Enable Firebase Authentication and set up phone number sign-in.
  3. Enable Firestore Database and Firebase Storage for storing user data and profile images.
  4. Download the google-services.json file and place it in your Android Studio project.
- **Android Studio Configuration**:
  - Ensure that all necessary dependencies (Firebase Auth, Firestore, Compose libraries) are added in build.gradle.
  - Sync the project with Gradle to confirm that all Firebase services are linked correctly.

b) Composable Functions and UI Design

- **Compose Components**:
  - Describe the use of composable functions for each part of the UI, such as login screens, profile screens, and chat screens.
  - Each screen is a collection of smaller composables: input fields, button components, and profile image placeholders.
- **State Management**:
  - Explain how state is managed across different screens using Compose's remember and MutableState to ensure smooth transitions and real-time updates.

c) Data Management and Real-Time Updates

- **Firestore for Real-Time Data**:
  - Messages are stored in Firestore, with each chat having a dedicated document to track messages.
  - Each message is updated in real-time, leveraging Firestore's snapshot listeners to reflect changes immediately in the chat interface.
- **Profile Information Storage**:
  - User profile data, including the display name and profile picture URL, is stored in Firestore.
  - When users log in, their profile data is retrieved and displayed, allowing for a personalized chat experience.

4. Implementation Process:

1. **Set Up Firebase Project and Android Studio Integration**
   - Walk through the process of creating a Firebase project, enabling the necessary services (Auth, Firestore, Storage), and linking it with your Android project.
2. **Implement Mobile Number Authentication**
   - Detail the code used to verify phone numbers, handle OTP verification, and manage authentication states.

3. **Design User Interface Using Compose UI**

   - Explain how each screen is structured, including composable functions for login, profile setup, and chat.
   - Discuss how navigation is managed using Compose's NavHost.

4. **Implement Profile Setup and Storage**

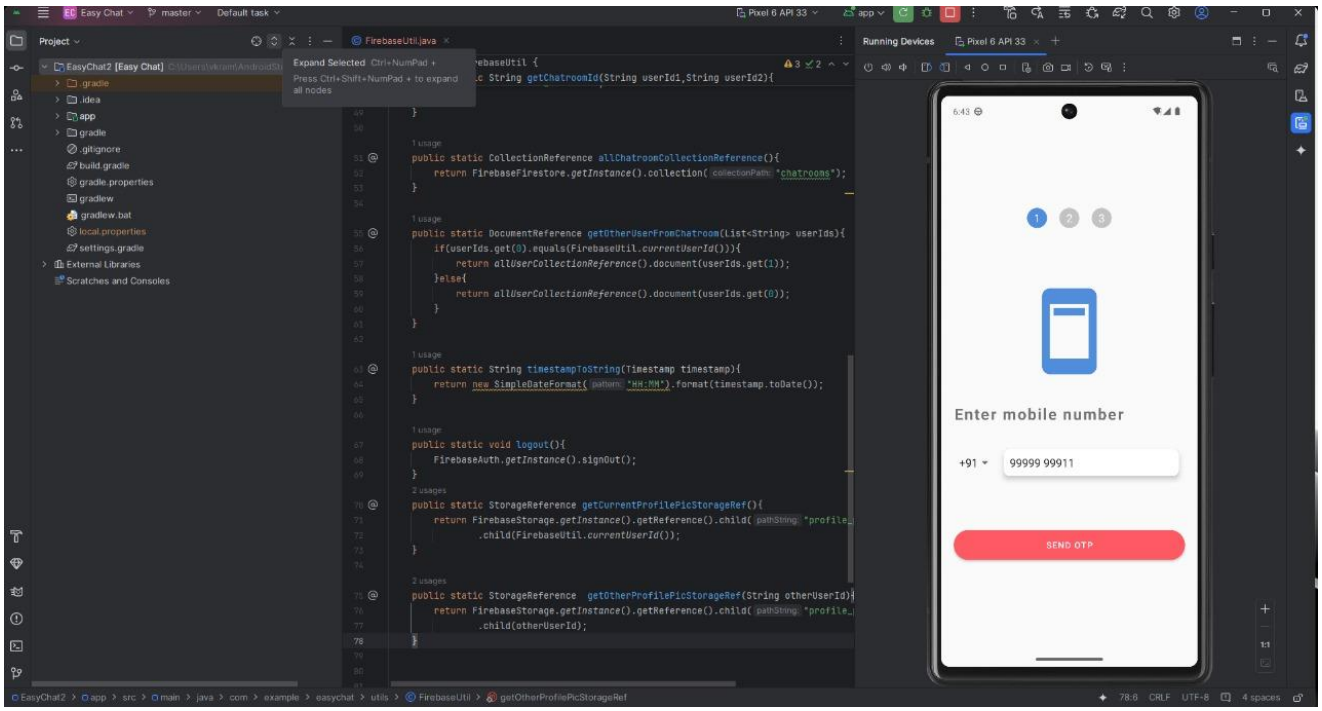
   - Describe the process for selecting and uploading profile images, saving metadata in Firestore, and displaying the user profile.

5. **Develop Chat Interface and Message Handling**

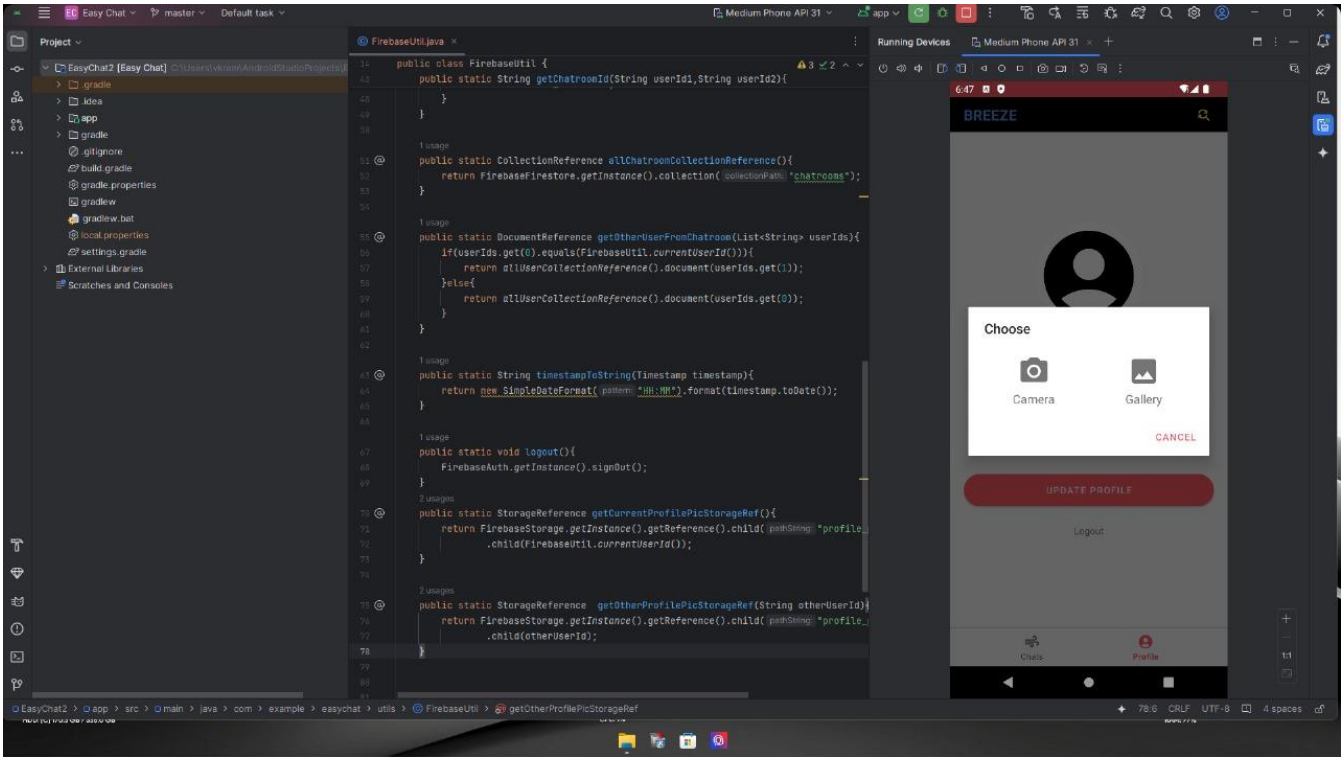
   - Describe how Firestore is used for real-time chat, how messages are stored and retrieved, and how Compose handles message display.

5. Output Screenshots:

Login Screen (with mobile number authentication)



Profile Setup Screen (with image upload and name entry)



Chat Screen (showcasing message exchange)