Assignment 3 - Simple Chat using Cloud Firestore – Due November 16th 2020 at 11:59 PM.
Submit your zipped Android Studio project. See syllabus for late penalties, etc. App must be written in Kotlin.

In this assignment, you will implement a simple messenger app (similar to Facebook messenger) in Android, using Firebase's 'Cloud Firestore' as the backend. You will also practice using Content Provider, and Android Authentication tools.

Figure 1 home screen

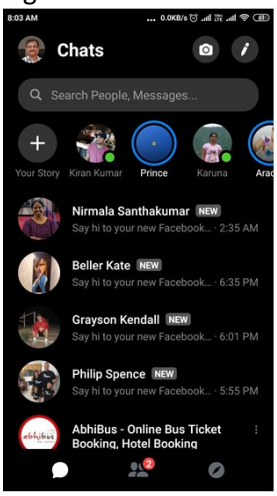
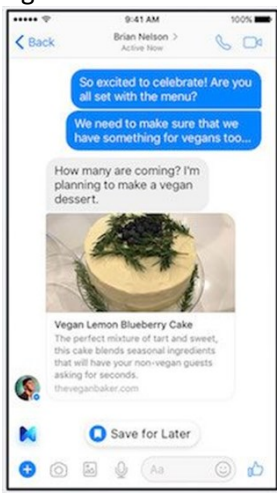


Figure 2 chat screen



Part 1 (30%): The home screen of 'Simple Chat' is a list of contacts that the user has stored in their phone. To populate this list, access Android's 'contacts provider' content provider to retrieve a list of contacts:
https://developer.android.com/training/contacts-provider/retrieve-names

If you're using an emulator, just add some dummy contacts to the device. Otherwise, test on your own device using your own contacts. The list that you retrieve from contacts provider must be displayed using a CursorAdapter that binds the list to a ListView.

Part 2 (30%): chat screen: when the user selects (taps on) one of the contacts from the home screen ListView, a new Activity is launched. This is the chat screen (see Figure 2). The user should be able to type messages in a text field at the bottom, and then press 'enter' to display the message in the display window above (see Figure 2 or any chat app). Put some effort into making this looks nice, and make sure, once enough messages have been entered, that the screen scrolls correctly (new messages should force older messages off the screen at the top, once the screen is full of messages).

Part 3 (30%): Cloud Firestore Integration: setup Cloud Firestore to work with your project (See lecture 11). When the user enters the chat screen with one of the contacts, set up a document or collection in Cloud Firestore to store the conversation. When the user enters a message in the chat screen, the message is uploaded to this document/collection. However, the app must also immediately subscribe to the document/collection set up in Cloud Firestore and listen for any changes. So, when the document/collection changes (due to message uploading), the user's app will be notified. Display a quick toast with a) first 5 characters of the message and b) time between pressing 'enter' to send the message, and the time your app's subscription to Cloud Firestore callback is executed.

- Basically, the toast should function as a 'ping' which tells the user how long it took to upload the message and then receive the notification that the message was received by Cloud Firestore.

Part 4 (10%): User Authentication - implement Email, Facebook, and Gmail sign in using FirebaseUI and Firebase Authentication. Create a rule in the Firebase database under "rules" tab that only allows people who have signed in to read/write from the database. Once the user has signed in (before entering the app), add them to the Cloud Firestore database that you created in (1) as a user. They will then be visible in the ListView from step 1. ONLY DO THIS IF STEPS 1-3 ARE WORKING PERFECTLY.

- Before attempting part 4, duplicate your Android Studio project. Submit to me two projects:
  - Project 1 – part 1-3 only
  - Project 2 – all parts (parts 1-4).
  - This is due to the fact that User Authentication caused some issues when marking in previous years.

https://firebase.google.com/docs/auth