Assignment- 5(B) Chat app with dual process

Objective:

Create basic chatting app with dual Processes

Description:

In this app I have used basic activities and xml file to create an interaction and user interface. I used Adapter at the server side to dynamically display incoming messages with list view. In addition to contract and database packages from previous assignment, I have created different packages like managers and content providers to design app more efficiently.

To send UDP packets from client to server I have used socket programming. In this I have created sockets at both the ends respectively. I have created a content provider that can be used to uniquely identify rows and tables from the database table. I have implemented Loader manager in order to enable the app to work in background rather than making it run on the main thread. I have also used entity managers in order to make all the CRUD operations to function asynchronously. In addition to previous assignment I have added a service package that which has 2 service classes that runs 2 different services in background so that the Application no longer freezes when the user clicks on next button. I have also implemented have two services: a bound service for handling the sending of messages, and an explicitly started and stopped service for handling receipt of messages.

Along with this I have also used content resolvers that handle all the connectivity with content provider. For SQLite database I have created adapter class which includes inner class that is used to create and update database. Also contract class is used to access underline database structure through book and author class. I have created 3 activities of which one is used to display peer list, second is used to change the client name as a preference and third one for displaying all the details related to that peer. Used join query to extract messages and peers and then displayed them in the third activity.

Conclusion:

In this way I have successfully created Chat App that can store, display users using Service and UI that run in 2 processes with 2 services.