Report of CS522 Assignment 8

Name: Yunfeng Wei
CWID: 10394963
E-mail: ywei14@stevens.edu

Part 1: Multi-Pane Navigation of Chat Rooms

1. Modify the database, create a new entity called Chatroom and create the ChatroomContract for the entity as Figure 1.1 and Figure 1.2. Modify the Content Provider and DatabaseHelper in order to query chatroom data.

```
public class Chatroom implements Parcelable {
    @Override
    public int describeContents() { return 0; }

    @Override
    public void writeToParcel(Parcel dest, int flags) {
        dest.writeLong(this.id);
        dest.writeString(this.name);
    }

    public static final Creator<Chatroom> CREATOR = new Creator<Chatroom>() {
        @Override
        public Chatroom createFromParcel(Parcel source) { return new Chatroom(source); }

        @Override
        public Chatroom[] newArray(int size) { return new Chatroom[size]; }
    };

    public long id;
    public String name;

    public Chatroom(String name) { this.name = name; }

    public Chatroom(long id, String name) {
        this.id = id;
        this.name = name;
    }

    public Chatroom(Parcel in) {
        this.id = in.readLong();
        this.name = in.readString();
    }

    public Chatroom(Cursor cursor) {
        this.id = ChatroomContract.getId(cursor);
        this.name = ChatroomContract.getName(cursor);
    }

    public void writeToProvider(ContentValues values) {
        ChatroomContract.setName(values, this.name);
    }
}
```

Figure 1.1

```java
public class ChatroomContract {
    public static final String ID = "_id";
    public static final String NAME = "chatroom";

    public static final String TABLE_NAME = "Chatrooms";
    public static final String CONTENT = "Chatroom";

    public static final Uri CONTENT_URI = MessageDbProvider.CONTENT_URI(MessageDbProvider.AUTHORITY, TABLE_NAME);

    public static Uri CONTENT_URI(String id) {
        return MessageDbProvider.withExtendedPath(CONTENT_URI, id);
    }

    public static String CONTENT_PATH = MessageDbProvider.CONTENT_PATH(CONTENT_URI); // Messages
    public static String CONTENT_ITEM_PATH = MessageDbProvider.CONTENT_PATH(CONTENT_URI("#")); // Messages/#

    public static long getId(Uri uri) { return Long.parseLong(uri.getLastPathSegment()); }
    public static long getId(Cursor cursor) {
        return cursor.getLong(cursor.getColumnIndexOrThrow(ID));
    }

    public static void setId(ContentValues values, long id) { values.put(ID, id); }

    public static String getName(Cursor cursor) {
        return cursor.getString(cursor.getColumnIndexOrThrow(NAME));
    }

    public static void setName(ContentValues values, String name) { values.put(NAME, name); }
}
```

Figure 1.2

2. Create a new Fragment called NavigationFragment which is used to show the list of chatrooms as Figure 2.1. When a chatroom name is clicked, the NavigationFragment will check if the device is in port mode or land mode, if it's in land mode, a new Fragment will show the

chatroom messages on the screen. Otherwise, it will start a new Activity to show the messages.

```
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
    // Inflate the layout for this fragment
    View rootView = inflater.inflate(R.layout.fragment_navigation, container, false);
    mainFrame = getActivity().findViewById(R.id.main_chatroom_container);
    final boolean MultiPane = getResources().getConfiguration().orientation == Configuration.ORIENTATION_LANDSCAPE;
    if (MultiPane) {
        Log.i(TAG, "Land mode");
    } else {
        Log.i(TAG, "Port mode");
    }
    manager = new ChatroomManager(getActivity(), (cursor) -> {
            return new Chatroom(cursor);
    }, NAVIGATION_FRAGMENT_LOADER_ID);
    listView = (ListView)rootView.findViewById(R.id.navigation_list);
    String[] from = new String[] {ChatroomContract.NAME};
    int[] to = new int[] {R.id.navigation_row_item};
    adapter = new SimpleCursorAdapter(getActivity(), R.layout.navigation_row, null, from, to, 0);
    listView.setAdapter(adapter);
    manager.QueryAsync(ChatroomContract.CONTENT_URI, new IQueryListener<Chatroom>() {
        @Override
        public void handleResults(TypedCursor<Chatroom> cursor) {
            Log.i(TAG, "Swap cursor");
            adapter.swapCursor(cursor.getCursor());
        }

        @Override
        public void closeResults() { adapter.swapCursor(null); }
    });

    listView.setChoiceMode(AbsListView.CHOICE_MODE_SINGLE);
    listView.setOnItemClickListener((parent, view, position, id) -> {
            Cursor cursor = adapter.getCursor();
            cursor.moveToPosition(position);
            Chatroom chatroom = new Chatroom(cursor);
            if (!MultiPane) { // Portrait mode
                Intent intent = new Intent(getActivity(), MainChatActivity.class);
                intent.putExtra(MainChatActivity.TAG, chatroom);
                startActivity(intent);
            } else {
                FragmentTransaction ft;
                Fragment fragment = new MainChatFragment();
                Bundle args = new Bundle();
                args.putParcelable(MainChatFragment.TAG, chatroom);
                fragment.setArguments(args);
```

Figure 2.1

3. Create a new Fragment called MainChatFragment which is used to show the chatroom messages on the screen as Figure 3.1. Furthermore, it also implements a new option menu called “SEND MESSAGE” to send a message and synchronize with the database as Figure 3.2.

```java
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
    // Inflate the layout for this fragment
    View rootView = inflater.inflate(R.layout.fragment_main_chat, container, false);
    messageList = (ListView)rootView.findViewById(R.id.message_list);
    backButton = (Button)rootView.findViewById(R.id.back_button);
    titleView = (TextView)rootView.findViewById(R.id.main_chatroom_title);

    String[] from = new String[] {MessageContract.MESSAGE_TEXT, ClientContract.NAME};
    int[] to = new int[] {R.id.message_row, R.id.sender_row};
    cursorAdapter = new SimpleCursorAdapter(getActivity(), R.layout.message_row, null, from, to, 0);

    messageList.setAdapter(cursorAdapter);

    chatroom = getArguments().getParcelable(TAG);
    titleView.setText(chatroom.name);
    MAIN_CHAT_FRAGMENT_LOADER_ID = (int)chatroom.id + 10;
    Log.i(TAG, "LOADER_ID in chatroom: " + String.valueOf(MAIN_CHAT_FRAGMENT_LOADER_ID));
    manager = new MessageManager(getActivity(), (cursor) -> {
            return new Message(cursor);
    }, MAIN_CHAT_FRAGMENT_LOADER_ID);

    manager.QueryAsync(MessageContract.CONTENT_URI, new String[] {chatroom.name}, new IQueryListener<Message>() {
        @Override
        public void handleResults(TypedCursor<Message> cursor) {
            cursorAdapter.swapCursor(cursor.getCursor());
        }

        @Override
        public void closeResults() { cursorAdapter.swapCursor(null); }
    });

    backButton.setOnClickListener((v) -> {
            View navigationView = getActivity().findViewById(R.id.main_navigation_drawer);
            boolean MultiPane = navigationView != null && navigationView.getVisibility() == View.VISIBLE;
            if (MultiPane) {
                FragmentTransaction ft = getFragmentManager().beginTransaction();
                //ft.replace(R.id.main_chatroom_container, new BlankFragment());
                ft.remove(MainChatFragment.this);
                ft.commit();
                getFragmentManager().executePendingTransactions();
            } else {
                getActivity().finish();
            }
```

Figure 3.1

```java
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    int id = item.getItemId();

    if (id == R.id.send) {
        if (ChatAppActivity.clientID < 0) {
            AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());
            builder.setTitle("Warning").setMessage("Can't send message, please register and sign in first!").setPositive
                    dialog.cancel();
            });
            builder.create().show();
        } else {
            SendDialogFragment.launch(getActivity(), TAG, chatroom);
        }
    }


    return super.onOptionsItemSelected(item);
}
```

Figure 3.2

4. Also create a new Activity called MainChatActivity as Figure 4.1, when in port mode, if a chatroom is clicked, the main activity will start this activity to show the messages, if the device returns to land mode, it will start the main activity and stop self.

```java
public class MainChatActivity extends ActionBarActivity implements IRegisterListener {
    public static final String TAG = MainChatActivity.class.getCanonicalName();

    public static final int MAIN_CHAT_ACTIVITY_LOADER_ID = 7;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main_chat);
        Chatroom chatroom = getIntent().getParcelableExtra(TAG);
        final boolean MultiPane = getResources().getConfiguration().orientation == Configuration.ORIENTATION_LANDSCAPE;
        if (MultiPane) {
            Intent intent = new Intent(this, ChatAppActivity.class);
            intent.putExtra(TAG, chatroom);
            startActivity(intent);
            finish();
        }
        Fragment fragment = new MainChatFragment();
        Bundle args = new Bundle();
        args.putParcelable(MainChatFragment.TAG, chatroom);
        fragment.setArguments(args);
        getFragmentManager().beginTransaction().replace(R.id.main_chatroom_container, fragment).commit();
        getFragmentManager().executePendingTransactions();
    }

    @Override
    public void register(String host, int port, String name) { return; }

    @Override
    public void createChatroom(String name) { return; }

    @Override
    public void send(Message message, Chatroom chatroom) {
        MessageManager manager = new MessageManager(this, (cursor) -> {
                return new Message(cursor);
        }, MAIN_CHAT_ACTIVITY_LOADER_ID);
        manager.persistAsync(message, ChatAppActivity.client, chatroom);
    }
}
```

Figure 4.1

5. Modify the main activity - ChatAppActivity - layout, separate into land mode and port mode. The land mode is shown as Figure 5.1, and the port mode is shown as Figure 5.2.

```xml
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
    android:layout_height="match_parent" tools:context=".ChatAppActivity" android:orientation="horizontal">
    <fragment
        android:layout_width="0sp"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:name="edu.stevens.cs522.simplecloudchatapp.Fragments.NavigationFragment"
        android:id="@+id/main_navigation_drawer"
        />
    <FrameLayout
        android:id="@+id/main_chatroom_container"
        android:layout_width="0sp"
        android:layout_height="match_parent"
        android:layout_weight="3" />
</LinearLayout>
```

Figure 5.1

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
    android:layout_height="match_parent" tools:context=".ChatAppActivity" android:orientation="horizontal">
    <fragment
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:name="edu.stevens.cs522.simplecloudchatapp.Fragments.NavigationFragment"
        android:id="@+id/main_navigation_drawer"
        />


</LinearLayout>
```

Figure 5.2

6. Modify the ChatAppActivity, implement a new interface called IRegisterListener, which is used to register client as Figure 6.1, create new chatroom as Figure 6.2 and send message as Figure 6.3. Furthermore, the activity also has options to create chatroom, register client and show clients as Figure 6.4.

```java
@Override
public void register(String host, int port, String name) {
    if (name.equals("")) {
        Toast.makeText(ChatAppActivity.this, "Register failed", Toast.LENGTH_SHORT).show();
        return;
    }
    Register register = new Register(host, port, name, registrationID);
    receiver = (IReceiver) (resultCode, resultData) -> {
            if (resultCode == RequestService.RESULT_REGISTER_OK) {
                reGetInfo();
                Toast.makeText(ChatAppActivity.this, "Register success", Toast.LENGTH_SHORT).show();
            } else {
                Toast.makeText(ChatAppActivity.this, "Register failed", Toast.LENGTH_SHORT).show();
            }
    };
    wrapper = new AckReceiverWrapper(new Handler());
    wrapper.setReceiver(receiver);
    serviceHelper.RegisterUser(register, wrapper);
}
```

Figure 6.1

```
@Override
public void createChatroom(String name) {
    Chatroom chatroom = new Chatroom(name);
    ChatroomManager manager = new ChatroomManager(this, (cursor) -> {
            return new Chatroom(cursor);
    }, CHAT_APP_LOADER_ID);
    manager.InsertAsync(chatroom, (chatroom) -> {
            if (chatroom == null) {
                AlertDialog.Builder builder = new AlertDialog.Builder(ChatAppActivity.this);
                builder.setTitle("Chatroom already exists!").setPositiveButton("OK", (dialog, which) -> {
                        dialog.cancel();
                });
                builder.create().show();
            } else {
                final boolean MultiPane = getResources().getConfiguration().orientation == Configuration.ORIENTATION_LAI
                if (!MultiPane) { // Portrait mode
                    Intent intent = new Intent(ChatAppActivity.this, MainChatActivity.class);
                    intent.putExtra(MainChatActivity.TAG, chatroom);
                    startActivity(intent);
                } else {
                    FragmentTransaction ft;
                    Fragment fragment = new MainChatFragment();
                    Bundle args = new Bundle();
                    args.putParcelable(MainChatFragment.TAG, chatroom);
                    fragment.setArguments(args);
                    ft = getFragmentManager().beginTransaction();
                    ft.replace(R.id.main_chatroom_container, fragment);
                    ft.commit();
                    getFragmentManager().executePendingTransactions();
                }
            }
    });
}
```

Figure 6.2

```
@Override
public void send(Message message, Chatroom chatroom) {
    MessageManager manager = new MessageManager(this, (cursor) -> {
            return new Message(cursor);
    }, CHAT_APP_LOADER_ID);
    manager.persistAsync(message, client, chatroom);
}
```

Figure 6.3

```java
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    // Handle action bar item clicks here. The action bar will
    // automatically handle clicks on the Home/Up button, so long
    // as you specify a parent activity in AndroidManifest.xml.
    int id = item.getItemId();

    //noinspection SimplifiableIfStatement
    if (id == R.id.action_settings) {
        SettingDialogFragment.launch(this, TAG);
    }

    if (id == R.id.clients) {
        Intent intent = new Intent(this, ClientsActivity.class);
        startActivity(intent);
        return true;
    }

    if (id == R.id.create_chatroom) {
        if (clientID < 0) {
            AlertDialog.Builder builder = new AlertDialog.Builder(this);
            builder.setTitle("Please register first!").setPositiveButton("OK", (dialog, which) -> {
                    dialog.cancel();
            });
            builder.create().show();
        } else {
            ChatroomCreateFragment.launch(this, TAG);
        }
    }

    return super.onOptionsItemSelected(item);
}
```

Figure 6.4

7. Create a new Fragment called ClientsFragment as Figure 7.1 which is used to show the clients as list and if a client is clicked, if the device is in port mode, it will start a new Activity to show the messages sent by the client, otherwise it will start a fragment to show the messages on the screen.

```java
@Override
public void onViewCreated(View view, Bundle savedInstanceState) {
    super.onViewCreated(view, savedInstanceState);
    manager = new ClientManager(getActivity(), (cursor) -> {
            return new Client(cursor);
    }, CLIENT_FRAGMENT_LOADER_ID);
    listView = (ListView)view.findViewById(R.id.client_fragment_list);
    String[] from = new String[] {ClientContract.NAME};
    int[] to = new int[] {R.id.client_list};
    cursorAdapter = new SimpleCursorAdapter(getActivity(), R.layout.client_list, null, from, to, 0);
    listView.setAdapter(cursorAdapter);
    manager.QueryAsync(ClientContract.CONTENT_URI, new IQueryListener<Client>() {
        @Override
        public void handleResults(TypedCursor<Client> cursor) {
            cursorAdapter.swapCursor(cursor.getCursor());
        }

        @Override
        public void closeResults() { cursorAdapter.swapCursor(null); }
    });
    listView.setChoiceMode(ListView.CHOICE_MODE_SINGLE);
    listView.setOnItemClickListener((parent, view, position, id) -> {
            Cursor cursor = cursorAdapter.getCursor();
            cursor.moveToPosition(position);
            Client client = new Client(cursor);
            client.id = ClientContract.getClientId(cursor);
            messageView = getActivity().findViewById(R.id.message_fragment);
            boolean isMultiPane = getResources().getConfiguration().orientation == Configuration.ORIENTATION_LANDSCAPE;
            if (isMultiPane) {
                Fragment fragment = new MessagesFragment();
                Bundle args = new Bundle();
                args.putParcelable(MessagesFragment.TAG, client);
                fragment.setArguments(args);
                getFragmentManager().beginTransaction().replace(R.id.message_fragment, fragment).commit();
                getFragmentManager().executePendingTransactions();
            } else {
                Intent intent = new Intent(getActivity(), MessagesActivity.class);
                intent.putExtra(ClientsActivity.CLIENT_ACTIVITY_KEY, client);
                startActivity(intent);
            }
    });
}
```

Figure 7.1

8. Create a new Fragment called MessagesFragment as Figure 8.1. This fragment will show all the messages and chatrooms sent by the client on the screen. Also create a new Activity called MessagesActivity as Figure 8.2 to show the messages if the device is in port mode.

```java
@Override
public void onViewCreated(View view, Bundle savedInstanceState) {
    super.onViewCreated(view, savedInstanceState);

    final Client client = getArguments().getParcelable(TAG);
    MESSAGES_FRAGMENT_LOADER_ID = (int)client.id + 10;

    nameView = (TextView)view.findViewById(R.id.messages_client_name);
    nameView.setText("Name: " + client.name);
    listView = (ListView)view.findViewById(R.id.messages_list);
    messageManager = new MessageManager(getActivity(), (cursor) -> {
            return new Message(cursor);
    }, MESSAGES_FRAGMENT_LOADER_ID);
    String[] from = new String[] {ChatroomContract.NAME, MessageContract.MESSAGE_TEXT};
    int[] to = new int[] {R.id.chatroom_client_row, R.id.messages_client_row};
    adapter = new SimpleCursorAdapter(getActivity(), R.layout.client_message_row, null, from, to, 0);
    listView.setAdapter(adapter);
    messageManager.QueryAsync(ClientContract.CONTENT_URI(String.valueOf(client.id)), new IQueryListener<Message>() {
        @Override
        public void handleResults(TypedCursor<Message> cursor) {
            adapter.swapCursor(cursor.getCursor());
            cursor.getCursor().setNotificationUri(getActivity().getContentResolver(), ClientContract.CONTENT_URI(String.
        }

        @Override
        public void closeResults() { adapter.swapCursor(null); }
    });

}
```

Figure 8.1

```java
public class MessagesActivity extends ActionBarActivity {
    public static final String TAG = MessagesActivity.class.getCanonicalName();

    @Override
    protected void onCreate(Bundle savedInstanceState) {


        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_messages);
        Intent intent = getIntent();
        final Client client = intent.getParcelableExtra(ClientsActivity.CLIENT_ACTIVITY_KEY);
        final boolean MultiPane = getResources().getConfiguration().orientation == Configuration.ORIENTATION_LANDSCAPE;
        if (MultiPane) {
            Intent back = new Intent(this, ClientsActivity.class);
            back.putExtra(TAG, client);
            startActivity(back);
            finish();
        }
        Fragment fragment = new MessagesFragment();
        Bundle args = new Bundle();
        args.putParcelable(MessagesFragment.TAG, client);
        fragment.setArguments(args);
        getFragmentManager().beginTransaction().replace(R.id.message_fragment, fragment).commit();
        getFragmentManager().executePendingTransactions();
    }

}
```

Figure 8.2

9. Separate the ClientsActivity layout into port mode as Figure 9.1 and land mode as Figure 9.2.

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent">
    <fragment
        android:layout_width="fill_parent"
        android:layout_height="match_parent"
        android:name="edu.stevens.cs522.simplecloudchatapp.Fragments.ClientsFragment"
        android:id="@+id/client_fragment" />

</LinearLayout>
```

Figure 9.1

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal" android:layout_width="match_parent"
    android:layout_height="match_parent">
    <fragment
        android:layout_width="0sp"
        android:layout_height="match_parent"
        android:id="@+id/client_fragment"
        android:layout_weight="1"
        android:name="edu.stevens.cs522.simplecloudchatapp.Fragments.ClientsFragment"
        />
    <FrameLayout
        android:layout_width="0sp"
        android:id="@+id/message_fragment"
        android:layout_height="match_parent"
        android:layout_weight="3" />
</LinearLayout>
```

Figure 9.2

Part 2. Dialogs for dialogues

1. Create a new DialogFragment called ChatroomCreateFragment as Figure 1.1. The fragment allows the user to create a new chatroom and insert the chatroom into database.

```
private IRegisterListener listener;

public static void launch(Activity context, String tag) {
    ChatroomCreateFragment dialog = new ChatroomCreateFragment();
    dialog.show(context.getFragmentManager(), tag);
}

@Override
public void onAttach(Activity activity) {
    super.onAttach(activity);
    if (!(activity instanceof IRegisterListener)) {
        throw new IllegalStateException("Activity mush implement IRegisterListener.");
    }
    listener = (IRegisterListener)activity;
}

public ChatroomCreateFragment() {
    // Required empty public constructor
}

@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
    // Inflate the layout for this fragment
    return inflater.inflate(R.layout.fragment_chatroom_create, container, false);
}

@Override
public void onViewCreated(View view, Bundle savedInstanceState) {
    super.onViewCreated(view, savedInstanceState);
    chatroomText = (EditText)view.findViewById(R.id.chatroom_name);
    chatroomText.setText("");
    createButton = (Button)view.findViewById(R.id.chatroom_create_button);
    cancelButton = (Button)view.findViewById(R.id.chatroom_cancel_button);
    cancelButton.setOnClickListener((v) -> {
            ChatroomCreateFragment.this.getDialog().cancel();
    });
    createButton.setOnClickListener((v) -> {
            String name = chatroomText.getText().toString();
            listener.createChatroom(name);
            ChatroomCreateFragment.this.getDialog().cancel();
    });
}
```

Figure 1.1

2. Create a new DialogFragment called SettingDialogFragment which is used to register a user to the cloud server as Figure 2.1.

```java
@Override
public void onAttach(Activity activity) {
    super.onAttach(activity);
    if (!(activity instanceof IRegisterListener)) {
        throw new IllegalStateException("Activity mush implement IRegisterListener.");
    }
    listener = (IRegisterListener)activity;
}

public static void launch(Activity context, String tag) {
    SettingDialogFragment fragment = new SettingDialogFragment();
    fragment.show(context.getFragmentManager(), tag);
}

@Nullable
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
    return inflater.inflate(R.layout.fragment_setting_dialog, container, false);
}

@Override
public void onViewCreated(View view, Bundle savedInstanceState) {
    super.onViewCreated(view, savedInstanceState);
    SharedPreferences sharedPreferences = getActivity().getSharedPreferences(SettingDialogFragment.MY_SHARED_PREF, getA
    textHost = (EditText)view.findViewById(R.id.destination_host);
    textPort = (EditText)view.findViewById(R.id.destination_port);
    textClientName = (EditText)view.findViewById(R.id.client_name);
    saveButton = (Button)view.findViewById(R.id.register_save_button);
    cancelButton = (Button)view.findViewById(R.id.register_cancel_button);
    textHost.setText(sharedPreferences.getString(PREF_HOST, DESTINATION_HOST_DEFAULT));
    textPort.setText(String.valueOf(sharedPreferences.getInt(PREF_PORT, DESTINATION_PORT_DEFAULT)));
    textClientName.setText(sharedPreferences.getString(PREF_USERNAME, CLIENT_NAME_DEFAULT));

    saveButton.setOnClickListener((v) -> {
            String host = textHost.getText().toString();
            int port = Integer.parseInt(textPort.getText().toString());
            String clientName = textClientName.getText().toString();
            listener.register(host, port, clientName);
            SettingDialogFragment.this.getDialog().cancel();
    });
    cancelButton.setOnClickListener((v) -> {
            SettingDialogFragment.this.getDialog().cancel();
    });
}
```

Figure 2.1

3. Create a new DialogFragment called SendDialogFragment. This dialog allows a user to send message into the chatroom and synchronize the message with the server as Figure 2.2.

```
public static void launch(Activity context, String tag, Chatroom chatroom) {
    SendDialogFragment dialog = new SendDialogFragment();
    Bundle args = new Bundle();
    args.putParcelable(TAG, chatroom);
    dialog.setArguments(args);
    dialog.show(context.getFragmentManager(), tag);

}

@Override
public void onAttach(Activity activity) {
    super.onAttach(activity);
    if (!(activity instanceof IRegisterListener)) {
        throw new IllegalStateException("Activity mush implement IRegisterListener.");
    }
    listener = (IRegisterListener)activity;
}

@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
    // Inflate the layout for this fragment
    return inflater.inflate(R.layout.fragment_send_dialog, container, false);
}


@Override
public void onViewCreated(View view, Bundle savedInstanceState) {
    super.onViewCreated(view, savedInstanceState);
    messageText = (EditText)view.findViewById(R.id.message_text);
    messageText.setText("");
    sendButton = (Button)view.findViewById(R.id.send_button);
    cancelButton = (Button)view.findViewById(R.id.send_cancel_button);
    cancelButton.setOnClickListener((v) -> {
            SendDialogFragment.this.getDialog().cancel();
    });
    sendButton.setOnClickListener((v) -> {
            String text = messageText.getText().toString();
            Message message = new Message(text, new Timestamp(new Date().getTime()));
            Chatroom chatroom = getArguments().getParcelable(TAG);
            listener.send(message, chatroom);
            SendDialogFragment.this.getDialog().cancel();
    });
}
```

Figure 3.1

4. Launch and test the App as the video shows.