WhatsApp Android Chat application

LAb-2

| comp-sci490-002: web/mobile programming| 04-12-2019

Team 11

Pavan kumar chongala class id: 09

Saivarun Gadhewar class id:13

Manohar Bollampally class id:07

Source CODE:

AndroidManifestation.xml

*<?***xml version="1.0" encoding="utf-8"***?>*<**manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.mys3soft.mys3chat"**>  
  
 <**uses-permission android:name="android.permission.INTERNET"** />  
 <**uses-permission android:name="android.permission.VIBRATE"** />  
 <**uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE"** />  
  
 <**application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="Whatsapp"  
 android:roundIcon="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 android:theme="@style/AppTheme"**>  
 <**activity android:name=".phonecall"**></**activity**>  
 <**activity  
 android:name=".ActivityMain"  
 android:label="Whatsapp"  
 android:theme="@style/AppTheme.NoActionBar"**>  
 <**intent-filter**>  
 <**action android:name="android.intent.action.MAIN"** />  
  
 <**category android:name="android.intent.category.LAUNCHER"** />  
 </**intent-filter**>  
 </**activity**>  
 <**activity  
 android:name=".ActivityLogin"  
 android:launchMode="singleTop"  
 android:noHistory="true"** />  
 <**activity android:name=".ActivityRegister"** />  
 <**activity android:name=".ActivityProfile"** />  
 <**activity  
 android:name=".ActivityChat"  
 android:launchMode="singleTop"  
 android:theme="@style/AppTheme.NoActionBar"** />  
 <**activity android:name=".ActivityAddContact"** />  
 <**activity android:name=".ActivityFriendProfile"** />  
 <**activity android:name=".ActivityNotifications"** />  
  
 <**service  
 android:name=".AppService"  
 android:enabled="true"  
 android:exported="true"  
 android:stopWithTask="false"** />  
  
 <**receiver  
 android:name=".AppReceiver"  
 android:enabled="true"  
 android:exported="true"**>  
 <**intent-filter**>  
 <**action android:name="com.mys3soft.mys3chat.restartservice"** />  
 </**intent-filter**>  
 </**receiver**>  
 </**application**>  
  
</**manifest**>

JAVA files:

Activitymain

**package** com.mys3soft.mys3chat;  
  
**import** android.app.Activity;  
**import** android.app.ProgressDialog;  
**import** android.content.DialogInterface;  
**import** android.content.Intent;  
**import** android.os.AsyncTask;  
**import** android.support.design.widget.TabLayout;  
**import** android.support.v7.app.AlertDialog;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.support.v7.widget.Toolbar;  
  
**import** android.support.v4.app.Fragment;  
**import** android.support.v4.app.FragmentManager;  
**import** android.support.v4.app.FragmentPagerAdapter;  
**import** android.support.v4.view.ViewPager;  
**import** android.os.Bundle;  
**import** android.text.InputType;  
**import** android.view.LayoutInflater;  
**import** android.view.Menu;  
**import** android.view.MenuItem;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
  
**import** android.widget.AdapterView;  
**import** android.widget.EditText;  
**import** android.widget.ListAdapter;  
**import** android.widget.ListView;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** com.firebase.client.Firebase;  
**import** com.mys3soft.mys3chat.Models.Message;  
**import** com.mys3soft.mys3chat.Models.StaticInfo;  
**import** com.mys3soft.mys3chat.Models.User;  
**import** com.mys3soft.mys3chat.Services.DataContext;  
**import** com.mys3soft.mys3chat.Services.IFireBaseAPI;  
**import** com.mys3soft.mys3chat.Services.LocalUserService;  
**import** com.mys3soft.mys3chat.Services.Tools;  
  
**import** org.json.JSONException;  
**import** org.json.JSONObject;  
  
**import** java.io.IOException;  
**import** java.text.DateFormat;  
**import** java.text.SimpleDateFormat;  
**import** java.util.ArrayList;  
**import** java.util.Date;  
**import** java.util.Iterator;  
**import** java.util.List;  
  
**import** retrofit2.Call;  
  
**public class** ActivityMain **extends** AppCompatActivity {  
  
 **private** SectionsPagerAdapter **mSectionsPagerAdapter**;  
 **private** ViewPager **mViewPager**;  
  
 **private** User **user**;  
 Firebase **refUser**;  
 **private** DataContext **db**;  
 **private** ProgressDialog **pd**;  
 **private** List<Message> **userLastChatList**;  
 **private** List<User> **userFriednList**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 Firebase.*setAndroidContext*(**this**);  
  
 **db** = **new** DataContext(**this**, **null**, **null**, 1);  
  
 **pd** = **new** ProgressDialog(**this**);  
 **pd**.setMessage(**"Refreshing..."**);  
  
 Toolbar toolbar = (Toolbar) findViewById(R.id.***toolbar***);  
 setSupportActionBar(toolbar);  
 **mSectionsPagerAdapter** = **new** SectionsPagerAdapter(getSupportFragmentManager());  
 *// Set up the ViewPager with the sections adapter.* **mViewPager** = (ViewPager) findViewById(R.id.***container***);  
 **mViewPager**.setAdapter(**mSectionsPagerAdapter**);  
 TabLayout tabLayout = (TabLayout) findViewById(R.id.***tabs***);  
 tabLayout.setupWithViewPager(**mViewPager**);  
  
 *// check if user exists in local db* **user** = LocalUserService.*getLocalUserFromPreferences*(**this**);  
 **if** (**user**.**Email** == **null**) {  
 *// send to activitylogin  
// Intent intent = new Intent(this, ActivityLogin.class);  
// startActivityForResult(intent, 100);  
//* } **else** {  
 startService(**new** Intent(**this**, AppService.**class**));  
 **if** (**refUser** == **null**) {  
 **refUser** = **new** Firebase(StaticInfo.*UsersURL* + **"/"** + **user**.**Email**);  
 }  
  
 }  
 }  
  
  
 @Override  
 **protected void** onStart() {  
 **super**.onStart();  
  
 *// check if user exists in local db* **user** = LocalUserService.*getLocalUserFromPreferences*(**this**);  
 **if** (**user**.**Email** == **null**) {  
 *// send to activitylogin* Intent intent = **new** Intent(**this**, ActivityLogin.**class**);  
 startActivityForResult(intent, 100);  
 **return**;  
 }  
  
 *// refresh last chat* **userLastChatList** = **db**.getUserLastChatList(**user**.**Email**);  
 ListAdapter lastChatAdp = **new** AdapterLastChat(**this**, **userLastChatList**);  
 **final** ListView lv\_LastChatList = (ListView) findViewById(R.id.***lv\_LastChatList***);  
 **if** (lv\_LastChatList != **null**) {  
 lv\_LastChatList.setAdapter(lastChatAdp);  
 *// reset listener* lv\_LastChatList.setOnItemLongClickListener(**new** AdapterView.OnItemLongClickListener() {  
 @Override  
 **public boolean** onItemLongClick(AdapterView<?> parent, View view, **int** position, **long** id) {  
 **if** (**userLastChatList**.size() <= position) **return false**;  
 **final** Message selectedMessageItem = **userLastChatList**.get(position);  
 **final** CharSequence options[] = **new** CharSequence[]{**"Delete Chat"**};  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(ActivityMain.**this**);  
 builder.setTitle(selectedMessageItem.**FriendFullName**);  
 builder.setItems(options, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** index) {  
 *// the user clicked on list[index]* **if** (index == 0) {  
 *// Delete Chat* **new** AlertDialog.Builder(ActivityMain.**this**)  
 .setTitle(selectedMessageItem.**FriendFullName**)  
 .setMessage(**"Are you sure to delete this chat?"**)  
 .setPositiveButton(**"Delete"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 **db**.deleteChat(**user**.**Email**, selectedMessageItem.**FromMail**);  
 Toast.*makeText*(getApplicationContext(), **"Chat deleted successfully"**, Toast.***LENGTH\_SHORT***).show();  
 **userLastChatList** = **db**.getUserLastChatList(**user**.**Email**);  
 ListAdapter adp = **new** AdapterLastChat(getApplicationContext(), **userLastChatList**);  
 lv\_LastChatList.setAdapter(adp);  
 }  
 })  
 .setNegativeButton(android.R.string.***no***, **null**)  
 .show();  
 }  
 }  
 });  
  
 builder.setNegativeButton(**"Cancel"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 dialog.cancel();  
 }  
 });  
 builder.show();  
 **return true**;  
 }  
 });  
  
 }  
  
 *// refresh contacts* **userFriednList** = **db**.getUserFriendList();  
 ListAdapter adp = **new** FriendListAdapter(**this**, **userFriednList**);  
 **final** ListView lv\_FriendList = (ListView) findViewById(R.id.***lv\_FriendList***);  
 **if** (lv\_FriendList != **null**) {  
 lv\_FriendList.setAdapter(adp);  
 lv\_FriendList.setOnItemLongClickListener(**new** AdapterView.OnItemLongClickListener() {  
 @Override  
 **public boolean** onItemLongClick(AdapterView<?> parent, View view, **int** position, **long** id) {  
 **if** (**userFriednList**.size() <= position) **return false**;  
 **final** User selectedUser = **userFriednList**.get(position);  
 **final** CharSequence options[] = **new** CharSequence[]{**"Profile"**, **"Delete Contact"**};  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(ActivityMain.**this**);  
 builder.setTitle(selectedUser.**FirstName** + **" "** + selectedUser.**LastName**);  
 builder.setItems(options, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** index) {  
 *// the user clicked on list[index]* **if** (index == 0) {  
 *// Profile* Intent intent = **new** Intent(ActivityMain.**this**, ActivityFriendProfile.**class**);  
 intent.putExtra(**"Email"**, selectedUser.**Email**);  
 startActivityForResult(intent, StaticInfo.*ChatAciviityRequestCode*);  
 } **else** {  
 *// Delete Contact* **new** AlertDialog.Builder(ActivityMain.**this**)  
 .setTitle(selectedUser.**FirstName** + **" "** + selectedUser.**LastName**)  
 .setMessage(**"Are you sure to delete this contact?"**)  
 .setPositiveButton(**"Delete"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 Firebase ref = **new** Firebase(StaticInfo.*EndPoint* + **"/friends/"** + **user**.**Email** + **"/"** + selectedUser.**Email**);  
 ref.removeValue();  
 *// delete from local database* **db**.deleteFriendByEmailFromLocalDB(selectedUser.**Email**);  
 Toast.*makeText*(ActivityMain.**this**, **"Contact deleted successfully"**, Toast.***LENGTH\_SHORT***).show();  
 **userFriednList** = **db**.getUserFriendList();  
 ListAdapter adp = **new** FriendListAdapter(ActivityMain.**this**, **userFriednList**);  
 lv\_FriendList.setAdapter(adp);  
 }  
 })  
 .setNegativeButton(android.R.string.***no***, **null**)  
 .show();  
 }  
 }  
 });  
  
 builder.setNegativeButton(**"Cancel"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 dialog.cancel();  
 }  
 });  
  
 builder.show();  
  
  
 **return true**;  
 }  
 });  
 }  
  
 *// set online status* **user** = LocalUserService.*getLocalUserFromPreferences*(**this**);  
 **if** (**user**.**Email** != **null**) {  
 **if** (**refUser** == **null**) {  
 **refUser** = **new** Firebase(StaticInfo.*UsersURL* + **"/"** + **user**.**Email**);  
 }  
 }  
 **if** (**refUser** != **null**)  
 **refUser**.child(**"Status"**).setValue(**"Online"**);  
  
 }  
  
 @Override  
 **protected void** onDestroy() {  
 **super**.onDestroy();  
 *// set last seen* DateFormat dateFormat = **new** SimpleDateFormat(**"dd MM yy hh:mm a"**);  
 Date date = **new** Date();  
 **refUser**.child(**"Status"**).setValue(dateFormat.format(date));  
 }  
  
 @Override  
 **public boolean** onCreateOptionsMenu(Menu menu) {  
 getMenuInflater().inflate(R.menu.***menu\_main***, menu);  
 **return true**;  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **int** id = item.getItemId();  
 **if** (id == R.id.***menu\_logout***) {  
 **new** AlertDialog.Builder(**this**)  
 .setTitle(**"Logout?"**)  
 .setMessage(**"Are you sure to logout, you will no longer receive notifications."**)  
 .setPositiveButton(**"Logout"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 *// set last seen* DateFormat dateFormat = **new** SimpleDateFormat(**"dd MM yy hh:mm a"**);  
 Date date = **new** Date();  
 **refUser**.child(**"Status"**).setValue(dateFormat.format(date));  
 **if** (LocalUserService.*deleteLocalUserFromPreferences*(getApplicationContext())) {  
 **db**.deleteAllFriendsFromLocalDB();  
 *// stopService(new Intent(getApplicationContext(), AppService.class));* Toast.*makeText*(getApplicationContext(), **"Logout Success"**, Toast.***LENGTH\_SHORT***).show();  
 Intent intent = **new** Intent(getApplicationContext(), ActivityLogin.**class**);  
 startActivityForResult(intent, 100);  
 } **else** {  
 Toast.*makeText*(getApplicationContext(), **"Logout Success"**, Toast.***LENGTH\_SHORT***).show();  
 Intent intent = **new** Intent(getApplicationContext(), ActivityLogin.**class**);  
 startActivityForResult(intent, 100);  
 }  
 }  
 })  
 .setNegativeButton(android.R.string.***no***, **null**)  
 .show();  
 **return true**;  
 }  
 **if** (id == R.id.***menu\_profile***) {  
 startActivity(**new** Intent(**this**, ActivityProfile.**class**));  
  
 }  
  
 **if** (id == R.id.***menu\_addContacts***) {  
 startActivity(**new** Intent(**this**, ActivityAddContact.**class**));  
 **return true**;  
 }  
  
 **if** (id == R.id.***menu\_notification***) {  
 startActivity(**new** Intent(**this**, ActivityNotifications.**class**));  
 **return true**;  
 }  
  
 **if** (id == R.id.***menu\_refresh***) {  
   
 **if** (Tools.*isNetworkAvailable*(**this**)){  
  
 FriendListTask t = **new** FriendListTask();  
 t.execute();  
 }**else** {  
 Toast.*makeText*(**this**, **"Please check your internet connection."**, Toast.***LENGTH\_SHORT***).show();  
 }  
   
 }  
  
 **return super**.onOptionsItemSelected(item);  
 }  
  
 **public static class** PlaceholderFragment **extends** Fragment {  
  
 **private static final** String ***ARG\_SECTION\_NUMBER*** = **"section\_number"**;  
 **private** View **rootView**;  
 **private** ListView **lv\_LastChatList**;  
 **private** DataContext **db**;  
 User **user**;  
 **private** List<User> **userFriendList**;  
 **private** List<Message> **userLastChatList**;  
  
  
 **public** PlaceholderFragment() {  
 }  
  
  
 **public static** PlaceholderFragment newInstance(**int** sectionNumber) {  
 PlaceholderFragment fragment = **new** PlaceholderFragment();  
 Bundle args = **new** Bundle();  
 args.putInt(***ARG\_SECTION\_NUMBER***, sectionNumber);  
 fragment.setArguments(args);  
 **return** fragment;  
 }  
  
 @Override  
 **public** View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 **if** (**user** == **null**) {  
 **user** = LocalUserService.*getLocalUserFromPreferences*(getActivity());  
 }  
 **db** = **new** DataContext(getActivity(), **null**, **null**, 1);  
 *// Chat tab* **if** (getArguments().getInt(***ARG\_SECTION\_NUMBER***) == 1) {  
 **rootView** = inflater.inflate(R.layout.***fragment\_chat***, container, **false**);  
 **userLastChatList** = **db**.getUserLastChatList(**user**.**Email**);  
 ListAdapter adp = **new** AdapterLastChat(getActivity(), **userLastChatList**);  
 **lv\_LastChatList** = (ListView) **rootView**.findViewById(R.id.***lv\_LastChatList***);  
 **lv\_LastChatList**.setAdapter(adp);  
 **lv\_LastChatList**.setOnItemClickListener(  
 **new** AdapterView.OnItemClickListener() {  
 @Override  
 **public void** onItemClick(AdapterView<?> parent, View view, **int** position, **long** id) {  
 TextView email = (TextView) view.findViewById(R.id.***tv\_lastChat\_HiddenEmail***);  
 TextView tv\_Name = (TextView) view.findViewById(R.id.***tv\_lastChat\_FriendFullName***);  
 Intent intend = **new** Intent(getActivity(), ActivityChat.**class**);  
 intend.putExtra(**"FriendEmail"**, email.getText().toString());  
 intend.putExtra(**"FriendFullName"**, tv\_Name.getText().toString());  
 startActivity(intend);  
 }  
 }  
 );  
  
 **lv\_LastChatList**.setOnItemLongClickListener(**new** AdapterView.OnItemLongClickListener() {  
 @Override  
 **public boolean** onItemLongClick(AdapterView<?> parent, View view, **int** position, **long** id) {  
 **if** (**userLastChatList**.size() <= position) **return false**;  
 **final** Message selectedMessageItem = **userLastChatList**.get(position);  
 **final** CharSequence options[] = **new** CharSequence[]{**"Delete Chat"**};  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(getActivity());  
 builder.setTitle(selectedMessageItem.**FriendFullName**);  
 builder.setItems(options, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** index) {  
 *// the user clicked on list[index]* **if** (index == 0) {  
 *// Delete Chat* **new** AlertDialog.Builder(getActivity())  
 .setTitle(selectedMessageItem.**FriendFullName**)  
 .setMessage(**"Are you sure to delete this chat?"**)  
 .setPositiveButton(**"Delete"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 **db**.deleteChat(**user**.**Email**, selectedMessageItem.**FromMail**);  
 Toast.*makeText*(getActivity(), **"Chat deleted successfully"**, Toast.***LENGTH\_SHORT***).show();  
 **userLastChatList** = **db**.getUserLastChatList(**user**.**Email**);  
 ListAdapter adp = **new** AdapterLastChat(getActivity(), **userLastChatList**);  
 **lv\_LastChatList** = (ListView) **rootView**.findViewById(R.id.***lv\_LastChatList***);  
 **lv\_LastChatList**.setAdapter(adp);  
 }  
 })  
 .setNegativeButton(android.R.string.***no***, **null**)  
 .show();  
 }  
 }  
 });  
  
 builder.setNegativeButton(**"Cancel"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 dialog.cancel();  
 }  
 });  
  
 builder.show();  
  
 **return true**;  
 }  
 });  
  
 **return rootView**;  
 }  
 *// Contacts tab* **else** {  
 **rootView** = inflater.inflate(R.layout.***fragment\_contact***, container, **false**);  
 **userFriendList** = **db**.getUserFriendList();  
 ListAdapter adp = **new** FriendListAdapter(getActivity(), **userFriendList**);  
 ListView lv\_FriendList = (ListView) **rootView**.findViewById(R.id.***lv\_FriendList***);  
 lv\_FriendList.setAdapter(adp);  
 lv\_FriendList.setOnItemClickListener(  
 **new** AdapterView.OnItemClickListener() {  
 @Override  
 **public void** onItemClick(AdapterView<?> parent, View view, **int** position, **long** id) {  
 TextView email = (TextView) view.findViewById(R.id.***tv\_HiddenEmail***);  
 TextView tv\_Name = (TextView) view.findViewById(R.id.***tv\_FriendFullName***);  
 Intent intend = **new** Intent(getActivity(), ActivityChat.**class**);  
 intend.putExtra(**"FriendEmail"**, email.getText().toString());  
 intend.putExtra(**"FriendFullName"**, tv\_Name.getText().toString());  
 startActivity(intend);  
 }  
 }  
  
 );  
  
 lv\_FriendList.setOnItemLongClickListener(**new** AdapterView.OnItemLongClickListener() {  
 @Override  
 **public boolean** onItemLongClick(AdapterView<?> parent, View view, **int** position, **long** id) {  
 **if** (**userFriendList**.size() <= position) **return false**;  
 **final** User selectedUser = **userFriendList**.get(position);  
 **final** CharSequence options[] = **new** CharSequence[]{**"Profile"**, **"Delete Contact"**};  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(getActivity());  
 builder.setTitle(selectedUser.**FirstName** + **" "** + selectedUser.**LastName**);  
 builder.setItems(options, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** index) {  
 *// the user clicked on list[index]* **if** (index == 0) {  
 *// Profile* Intent intent = **new** Intent(getActivity(), ActivityFriendProfile.**class**);  
 intent.putExtra(**"Email"**, selectedUser.**Email**);  
 startActivityForResult(intent, StaticInfo.*ChatAciviityRequestCode*);  
 } **else** {  
 *// Delete Contact* **new** AlertDialog.Builder(getActivity())  
 .setTitle(selectedUser.**FirstName** + **" "** + selectedUser.**LastName**)  
 .setMessage(**"Are you sure to delete this contact?"**)  
 .setPositiveButton(**"Delete"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 Firebase ref = **new** Firebase(StaticInfo.*EndPoint* + **"/friends/"** + **user**.**Email** + **"/"** + selectedUser.**Email**);  
 ref.removeValue();  
 *// delete from local database* **db**.deleteFriendByEmailFromLocalDB(selectedUser.**Email**);  
 Toast.*makeText*(getActivity(), **"Contact deleted successfully"**, Toast.***LENGTH\_SHORT***).show();  
 **userFriendList** = **db**.getUserFriendList();  
 ListAdapter adp = **new** FriendListAdapter(getActivity(), **userFriendList**);  
 ListView lv\_FriendList = (ListView) **rootView**.findViewById(R.id.***lv\_FriendList***);  
 lv\_FriendList.setAdapter(adp);  
 }  
 })  
 .setNegativeButton(android.R.string.***no***, **null**)  
 .show();  
 }  
 }  
 });  
  
 builder.setNegativeButton(**"Cancel"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 dialog.cancel();  
 }  
 });  
  
 builder.show();  
  
  
 **return true**;  
 }  
 });  
 **return rootView**;  
 }  
  
 }  
 }  
  
  
 **public class** SectionsPagerAdapter **extends** FragmentPagerAdapter {  
  
  
 **public** SectionsPagerAdapter(FragmentManager fm) {  
 **super**(fm);  
 }  
  
 @Override  
 **public** Fragment getItem(**int** position) {  
 **return** PlaceholderFragment.*newInstance*(position + 1);  
 }  
  
 @Override  
 **public int** getCount() {  
 **return** 2;  
 }  
  
 @Override  
 **public** CharSequence getPageTitle(**int** position) {  
 **switch** (position) {  
 **case** 0:  
 **return "CHATS"**;  
 **case** 1:  
 **return "CONTACTS"**;  
 }  
 **return null**;  
 }  
 }  
  
  
 **public class** FriendListTask **extends** AsyncTask<Void, Void, String> {  
  
 @Override  
 **protected void** onPreExecute() {  
 **pd**.show();  
 }  
  
 @Override  
 **protected** String doInBackground(Void... params) {  
 **user** = LocalUserService.*getLocalUserFromPreferences*(getApplicationContext());  
 IFireBaseAPI api = Tools.*makeRetroFitApi*();  
 Call<String> call = api.getUserFriendsListAsJsonString(StaticInfo.*FriendsURL* + **"/"** + **user**.**Email** + **".json"**);  
 **try** {  
 **return** call.execute().body();  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 **pd**.hide();  
 }  
 **return null**;  
 }  
  
 @Override  
 **protected void** onPostExecute(String jsonListString) {  
  
 **try** {  
 **user** = LocalUserService.*getLocalUserFromPreferences*(getApplicationContext());  
 List<User> friendList = **new** ArrayList<>();  
 JSONObject userFriendTree = **new** JSONObject(jsonListString);  
 **for** (Iterator iterator = userFriendTree.keys(); iterator.hasNext(); ) {  
 String key = (String) iterator.next();  
 User friend = **new** User();  
 JSONObject friendJson = userFriendTree.getJSONObject(key);  
 friend.**Email** = friendJson.getString(**"Email"**);  
 friend.**FirstName** = friendJson.getString(**"FirstName"**);  
 friend.**LastName** = friendJson.getString(**"LastName"**);  
 friendList.add(friend);  
 }  
  
 *// refresh local database* **db** = **new** DataContext(getApplicationContext(), **null**, **null**, 1);  
 **db**.refreshUserFriendList(friendList);  
  
 *// set to adapter* ListAdapter adp = **new** FriendListAdapter(getApplicationContext(), **db**.getUserFriendList());  
 ListView lv\_FriendList = (ListView) findViewById(R.id.***lv\_FriendList***);  
 lv\_FriendList.setAdapter(adp);  
 **pd**.hide();  
 Toast.*makeText*(ActivityMain.**this**, **"Contact list is updated"**, Toast.***LENGTH\_SHORT***).show();  
 } **catch** (JSONException e) {  
 **pd**.hide();  
 e.printStackTrace();  
 }  
 }  
 }  
  
  
 @Override  
 **protected void** onActivityResult(**int** requestCode, **int** resultCode, Intent data) {  
 **super**.onActivityResult(requestCode, resultCode, data);  
 **user** = LocalUserService.*getLocalUserFromPreferences*(**this**);  
 **if** (requestCode == 100 && resultCode == Activity.***RESULT\_OK***) {  
 **if** (**refUser** == **null**) {  
 **refUser** = **new** Firebase(StaticInfo.*UsersURL* + **"/"** + **user**.**Email**);  
 }  
 startService(**new** Intent(getApplicationContext(), AppService.**class**));  
 FriendListTask t = **new** FriendListTask();  
 t.execute();  
 }  
 }  
  
  
}

ActivityLogin:

**package** com.mys3soft.mys3chat;  
  
**import** android.app.Activity;  
**import** android.app.ProgressDialog;  
**import** android.content.Intent;  
**import** android.content.SharedPreferences;  
**import** android.os.AsyncTask;  
**import** android.provider.ContactsContract;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.Toast;  
  
**import** com.firebase.client.Firebase;  
**import** com.mys3soft.mys3chat.Models.StaticInfo;  
**import** com.mys3soft.mys3chat.Models.User;  
**import** com.mys3soft.mys3chat.Services.DataContext;  
**import** com.mys3soft.mys3chat.Services.IFireBaseAPI;  
**import** com.mys3soft.mys3chat.Services.Tools;  
  
**import** org.json.JSONException;  
**import** org.json.JSONObject;  
  
**import** java.io.IOException;  
**import** java.util.List;  
  
**import** retrofit2.Call;  
**import** retrofit2.Retrofit;  
**import** retrofit2.converter.scalars.ScalarsConverterFactory;  
  
  
**public class** ActivityLogin **extends** AppCompatActivity {  
  
 DataContext **db** = **new** DataContext(**this**, **null**, **null**, 1);  
  
  
 EditText **et\_Email**, **et\_Password**;  
 Button **btn\_Login**;  
 **public static final** String ***ENDPOINT*** = **"https://mys3chat.firebaseio.com"**;  
 ProgressDialog **pd**;  
 String **email**;  
  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_login***);  
 **et\_Email** = (EditText) findViewById(R.id.***et\_Email***);  
 **et\_Password** = (EditText) findViewById(R.id.***et\_Password***);  
 }  
  
  
 **public void** btnLoginClick(View view) {  
 **if** (!Tools.*isNetworkAvailable*(**this**)) {  
 Toast.*makeText*(**this**, **"Please check your internet connection."**, Toast.***LENGTH\_SHORT***).show();  
 } **else if** (**et\_Email**.getText().toString().equals(**""**)) {  
 **et\_Email**.setError(**"Email cannot be empty"**);  
  
 } **else if** (**et\_Password**.getText().toString().equals(**""**)) {  
 **et\_Password**.setError(**"Password cannot be empty"**);  
 } **else** {  
 **pd** = **new** ProgressDialog(**this**);  
 **pd**.setMessage(**"Loading..."**);  
 **pd**.show();  
 **email** = Tools.*encodeString*(**et\_Email**.getText().toString());  
 LoginTask t = **new** LoginTask();  
 t.execute();  
 }  
 }  
  
  
 **public class** LoginTask **extends** AsyncTask<Void, Void, String> {  
 @Override  
 **protected** String doInBackground(Void... params) {  
 Retrofit retrofit = **new** Retrofit.Builder()  
 .baseUrl(***ENDPOINT***)  
 .addConverterFactory(ScalarsConverterFactory.*create*())  
 .build();  
  
 IFireBaseAPI api = retrofit.create(IFireBaseAPI.**class**);  
 *// Call<String> call = api.getAllUsersAsJsonString();* Call<String> call = api.getSingleUserByEmail(StaticInfo.*UsersURL* + **"/"** + **email** + **".json"**);  
 **try** {  
 **return** call.execute().body();  
 } **catch** (Exception e) {  
 **pd**.hide();  
 **return "null"**;  
 }  
 }  
  
 @Override  
 **protected void** onPostExecute(String jsonString) {  
 **try** {  
 **if** (!jsonString.trim().equals(**"null"**)) {  
 JSONObject userObj = **new** JSONObject(jsonString);  
 String pass = **et\_Password**.getText().toString();  
 **if** (userObj.getString(**"Password"**).equals(pass)) {  
 **pd**.hide();  
 SharedPreferences pref = getApplicationContext().getSharedPreferences(**"LocalUser"**, 0);  
 SharedPreferences.Editor editor = pref.edit();  
 editor.putString(**"Email"**, **email**);  
 editor.putString(**"FirstName"**, userObj.getString(**"FirstName"**));  
 editor.putString(**"LastName"**, userObj.getString(**"LastName"**));  
 editor.commit();  
 setResult(Activity.***RESULT\_OK***);  
 finish();  
 } **else** {  
 **pd**.hide();  
 Toast.*makeText*(ActivityLogin.**this**, **"Incorecct email or password"**, Toast.***LENGTH\_LONG***).show();  
 }  
 } **else** {  
 **pd**.hide();  
 Toast.*makeText*(ActivityLogin.**this**, **"Incorecct email or password"**, Toast.***LENGTH\_LONG***).show();  
 }  
  
 } **catch** (JSONException e) {  
 **pd**.hide();  
 e.printStackTrace();  
 }  
  
 }  
  
 }  
  
  
 **public void** btnSignUpClick(View view) {  
  
 startActivity(**new** Intent(**this**, ActivityRegister.**class**));  
 }  
  
  
 @Override  
 **public void** onBackPressed() {  
 Intent intent = **new** Intent(Intent.***ACTION\_MAIN***);  
 intent.addCategory(Intent.***CATEGORY\_HOME***);  
 intent.setFlags(Intent.***FLAG\_ACTIVITY\_NEW\_TASK***);  
 startActivity(intent);  
 }  
}

Activityregister

**package** com.mys3soft.mys3chat;  
  
**import** android.app.ProgressDialog;  
**import** android.os.AsyncTask;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.ListAdapter;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** com.firebase.client.ChildEventListener;  
**import** com.firebase.client.DataSnapshot;  
**import** com.firebase.client.Firebase;  
**import** com.firebase.client.FirebaseError;  
**import** com.mys3soft.mys3chat.Models.StaticInfo;  
**import** com.mys3soft.mys3chat.Models.User;  
**import** com.mys3soft.mys3chat.Services.DataContext;  
**import** com.mys3soft.mys3chat.Services.IFireBaseAPI;  
**import** com.mys3soft.mys3chat.Services.LocalUserService;  
**import** com.mys3soft.mys3chat.Services.Tools;  
  
**import** org.json.JSONException;  
**import** org.json.JSONObject;  
  
**import** java.io.IOException;  
**import** java.text.DateFormat;  
**import** java.text.SimpleDateFormat;  
**import** java.util.ArrayList;  
**import** java.util.Date;  
**import** java.util.Iterator;  
**import** java.util.List;  
  
**import** retrofit2.Call;  
  
**public class** ActivityRegister **extends** AppCompatActivity {  
  
  
 EditText **et\_Email**, **et\_Password**, **et\_FirstName**, **et\_LastName**;  
 Button **btn\_Register**;  
 ProgressDialog **pd**;  
 String **email**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_register***);  
  
 Firebase.*setAndroidContext*(**this**);  
  
 **pd** = **new** ProgressDialog(**this**);  
 **pd**.setMessage(**"Loading..."**);  
  
 **et\_Email** = (EditText) findViewById(R.id.***et\_Email\_Rigister***);  
 **et\_Password** = (EditText) findViewById(R.id.***et\_Password\_Rigister***);  
 **et\_FirstName** = (EditText) findViewById(R.id.***et\_FirstName\_Rigister***);  
 **et\_LastName** = (EditText) findViewById(R.id.***et\_LastName\_Rigister***);  
  
 }  
  
 **public void** btn\_RegClick(View view) {  
 **if** (!Tools.*isNetworkAvailable*(**this**)) {  
 Toast.*makeText*(**this**, **"Please check your internet connection."**, Toast.***LENGTH\_SHORT***).show();  
 } **else if** (**et\_FirstName**.getText().toString().equals(**""**)) {  
 **et\_FirstName**.setError(**"Enter Firstname"**);  
 } **else if** (**et\_LastName**.getText().toString().equals(**""**)) {  
 **et\_LastName**.setError(**"Enter Lastname"**);  
 } **else if** (**et\_Email**.getText().toString().equals(**""**) || !Tools.*isValidEmail*(**et\_Email**.getText().toString())) {  
 **et\_Email**.setError(**"Enter Valid Email"**);  
 } **else if** (**et\_Password**.getText().toString().equals(**""**)) {  
 **et\_Password**.setError(**"Enter Password"**);  
 } **else** {  
 **email** = Tools.*encodeString*(**et\_Email**.getText().toString());  
 RegisterUserTask t = **new** RegisterUserTask();  
 t.execute();  
 }  
 }  
  
 **public class** RegisterUserTask **extends** AsyncTask<Void, Void, String> {  
  
 @Override  
 **protected void** onPreExecute() {  
 **pd**.show();  
 }  
  
 @Override  
 **protected** String doInBackground(Void... params) {  
  
 IFireBaseAPI api = Tools.*makeRetroFitApi*();  
 Call<String> call = api.getSingleUserByEmail(StaticInfo.*UsersURL* + **"/"** + **email** + **".json"**);  
 **try** {  
 **return** call.execute().body();  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 **pd**.hide();  
 }  
 **return null**;  
 }  
  
 @Override  
 **protected void** onPostExecute(String jsonString) {  
 **try** {  
 **if** (jsonString.trim().equals(**"null"**)) {  
 Firebase firebase = **new** Firebase(StaticInfo.*UsersURL*);  
 firebase.child(**email**).child(**"FirstName"**).setValue(**et\_FirstName**.getText().toString());  
 firebase.child(**email**).child(**"LastName"**).setValue(**et\_LastName**.getText().toString());  
 firebase.child(**email**).child(**"Email"**).setValue(**email**);  
 firebase.child(**email**).child(**"Password"**).setValue(**et\_Password**.getText().toString());  
 DateFormat dateFormat = **new** SimpleDateFormat(**"dd MM yy hh:mm a"**);  
 Date date = **new** Date();  
 firebase.child(**email**).child(**"Status"**).setValue(dateFormat.format(date));  
 Toast.*makeText*(getApplicationContext(), **"Signup Success"**, Toast.***LENGTH\_SHORT***).show();  
 **pd**.hide();  
 finish();  
 } **else** {  
 Toast.*makeText*(getApplicationContext(), **"Email already exists"**, Toast.***LENGTH\_SHORT***).show();  
 **pd**.hide();  
 }  
 } **catch** (Exception e) {  
 **pd**.hide();  
 e.printStackTrace();  
 }  
 }  
 }  
  
  
}

ActivityAddcontacts

**package** com.mys3soft.mys3chat;  
  
**import** android.app.ProgressDialog;  
**import** android.content.Intent;  
**import** android.os.AsyncTask;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.AdapterView;  
**import** android.widget.EditText;  
**import** android.widget.ListAdapter;  
**import** android.widget.ListView;  
**import** android.widget.ProgressBar;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** com.mys3soft.mys3chat.Models.User;  
**import** com.mys3soft.mys3chat.Services.IFireBaseAPI;  
**import** com.mys3soft.mys3chat.Services.LocalUserService;  
**import** com.mys3soft.mys3chat.Services.Tools;  
  
**import** org.json.JSONException;  
**import** org.json.JSONObject;  
  
**import** java.io.IOException;  
**import** java.util.ArrayList;  
**import** java.util.Iterator;  
**import** java.util.List;  
  
**import** retrofit2.Call;  
  
**public class** ActivityAddContact **extends** AppCompatActivity {  
  
  
 ListView **lv\_SerachList**;  
 EditText **searchKey**;  
 ProgressDialog **pd**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_add\_contact***);  
 **pd** = **new** ProgressDialog(**this**);  
 **pd**.setMessage(**"Searching..."**);  
 **lv\_SerachList** = (ListView) findViewById(R.id.***lv\_AddContactList***);  
 **searchKey** = (EditText) findViewById(R.id.***et\_SearchKey***);  
 *// listener for item click* **lv\_SerachList**.setOnItemClickListener(  
 **new** AdapterView.OnItemClickListener() {  
 @Override  
 **public void** onItemClick(AdapterView<?> parent, View view, **int** position, **long** id) {  
 TextView email = (TextView) view.findViewById(R.id.***tv\_HiddenEmail***);  
 *// start FriendProfileFull* Intent intent = **new** Intent(ActivityAddContact.**this**, ActivityFriendProfile.**class**);  
 intent.putExtra(**"Email"**, email.getText().toString());  
 startActivity(intent);  
 }  
 }  
 );  
 }  
  
 **public void** btn\_SearchClick(View view) {  
 **if** (!**searchKey**.getText().toString().trim().equals(**""**) && **searchKey**.getText().toString().length() > 2) {  
  
 **if** (Tools.*isNetworkAvailable*(**this**)){  
 FindFriendsTask t = **new** FindFriendsTask();  
 t.execute();  
 }**else** {  
 Toast.*makeText*(**this**, **"Please check your internet connection."**, Toast.***LENGTH\_SHORT***).show();  
 }  
  
 } **else** {  
 **searchKey**.setText(**""**);  
 Toast.*makeText*(**this**, **"Input at least 3 characters"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
  
 **public class** FindFriendsTask **extends** AsyncTask<Void, Void, String> {  
  
 @Override  
 **protected void** onPreExecute() {  
 **pd**.show();  
 }  
  
 @Override  
 **protected** String doInBackground(Void... params) {  
 IFireBaseAPI api = Tools.*makeRetroFitApi*();  
 Call<String> call = api.getAllUsersAsJsonString();  
 **try** {  
 **return** call.execute().body();  
 } **catch** (IOException e) {  
 **pd**.hide();  
 Toast.*makeText*(ActivityAddContact.**this**, **"Please check your internet connection."**, Toast.***LENGTH\_SHORT***).show();  
 }  
 **return null**;  
 }  
  
  
 @Override  
 **protected void** onPostExecute(String jsonListString) {  
  
 **try** {  
 User user = LocalUserService.*getLocalUserFromPreferences*(getApplicationContext());  
 JSONObject jsonObjectList = **new** JSONObject(jsonListString);  
 List<User> friendList = **new** ArrayList<>();  
 **for** (Iterator iterator = jsonObjectList.keys(); iterator.hasNext(); ) {  
 **try** {  
 String key = (String) iterator.next();  
 JSONObject item = jsonObjectList.getJSONObject(key);  
 User f = **new** User();  
 f.**Email** = item.getString(**"Email"**);  
 f.**FirstName** = item.getString(**"FirstName"**);  
 f.**LastName** = item.getString(**"LastName"**);  
 String serKey = Tools.*encodeString*(**searchKey**.getText().toString()).toLowerCase().trim();  
 String fullName = f.**FirstName**.toLowerCase() + **" "** + f.**LastName**.toLowerCase();  
 **if** (f.**Email**.toLowerCase().contains(serKey) || fullName.contains(serKey)) {  
 **if** (!f.**Email**.equals(user.**Email**)) {  
 friendList.add(f);  
 }  
 }  
 } **catch** (Exception exx) {  
 **continue**;  
 }  
 }  
 ListAdapter adp = **new** FriendListAdapter(ActivityAddContact.**this**, friendList);  
 **lv\_SerachList**.setAdapter(adp);  
 **pd**.hide();  
  
 } **catch** (JSONException e) {  
 **pd**.hide();  
 e.printStackTrace();  
 }  
  
  
 }  
 }  
  
  
  
}

ActivityChat

**package** com.mys3soft.mys3chat;  
  
  
**import** android.app.Activity;  
**import** android.content.DialogInterface;  
**import** android.content.Intent;  
**import** android.content.res.ColorStateList;  
**import** android.graphics.Color;  
**import** android.support.design.widget.FloatingActionButton;  
**import** android.support.v4.widget.SwipeRefreshLayout;  
**import** android.support.v7.app.AlertDialog;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.support.v7.widget.Toolbar;  
**import** android.text.Editable;  
**import** android.text.SpannableString;  
**import** android.text.TextWatcher;  
**import** android.text.style.ForegroundColorSpan;  
**import** android.text.style.RelativeSizeSpan;  
**import** android.view.Gravity;  
**import** android.view.Menu;  
**import** android.view.MenuItem;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.EditText;  
**import** android.widget.ImageView;  
**import** android.widget.LinearLayout;  
**import** android.widget.ScrollView;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** com.firebase.client.ChildEventListener;  
**import** com.firebase.client.DataSnapshot;  
**import** com.firebase.client.Firebase;  
**import** com.firebase.client.FirebaseError;  
**import** com.mys3soft.mys3chat.Models.Message;  
**import** com.mys3soft.mys3chat.Models.StaticInfo;  
**import** com.mys3soft.mys3chat.Models.User;  
**import** com.mys3soft.mys3chat.Services.DataContext;  
**import** com.mys3soft.mys3chat.Services.LocalUserService;  
**import** com.mys3soft.mys3chat.Services.Tools;  
  
**import** java.text.DateFormat;  
**import** java.text.SimpleDateFormat;  
**import** java.util.Date;  
**import** java.util.HashMap;  
**import** java.util.List;  
**import** java.util.Map;  
  
**import** hani.momanii.supernova\_emoji\_library.Actions.EmojIconActions;  
**import** hani.momanii.supernova\_emoji\_library.Helper.EmojiconEditText;  
**import** hani.momanii.supernova\_emoji\_library.Helper.EmojiconTextView;  
  
  
**public class** ActivityChat **extends** AppCompatActivity {  
 DataContext **db** = **new** DataContext(**this**, **null**, **null**, 1);  
 EditText **messageArea**;  
 ScrollView **scrollView**;  
 LinearLayout **layout**;  
 Firebase **reference1**, **reference2**, **refNotMess**, **refFriend**;  
 User **user**;  
 String **friendEmail**;  
 Firebase **refUser**;  
 **private int pageNo** = 2;  
 **private** FloatingActionButton **submit\_btn**;  
  
 **private** ChildEventListener **reference1Listener**;  
 **private** ChildEventListener **refFriendListener**;  
 **private** String **friendFullName** = **""**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_chat***);  
 Toolbar toolbar = (Toolbar) findViewById(R.id.***toolbarChatActivity***);  
 setSupportActionBar(toolbar);  
 **messageArea** = (EditText) findViewById(R.id.***et\_Message***);  
 **scrollView** = (ScrollView) findViewById(R.id.***scrollView***);  
 **layout** = (LinearLayout) findViewById(R.id.***layout1***);  
 **user** = LocalUserService.*getLocalUserFromPreferences*(**this**);  
 Firebase.*setAndroidContext*(**this**);  
 **reference1Listener** = **new** ChildEventListener() {  
 @Override  
 **public void** onChildAdded(DataSnapshot dataSnapshot, String s) {  
  
 **if** (!dataSnapshot.getKey().equals(StaticInfo.*TypingStatus*)) {  
 Map map = dataSnapshot.getValue(Map.**class**);  
 String mess = map.get(**"Message"**).toString();  
 String senderEmail = map.get(**"SenderEmail"**).toString();  
 String sentDate = map.get(**"SentDate"**).toString();  
 **try** {  
 *// remove from server* **reference1**.child(dataSnapshot.getKey()).removeValue();  
 *// save message on local db* **db**.saveMessageOnLocakDB(senderEmail, **user**.**Email**, mess, sentDate);  
 **if** (senderEmail.equals(**user**.**Email**)) {  
 *// login user* appendMessage(mess, sentDate, 1, **false**);  
 } **else** {  
 appendMessage(mess, sentDate, 2, **false**);  
 }  
 } **catch** (Exception e) {  
  
 }  
 } **else** {  
 *// show typing status* String typingStatus = dataSnapshot.getValue().toString();  
 **if** (typingStatus.equals(**"Typing"**)) {  
 getSupportActionBar().setSubtitle(typingStatus + **"..."**);  
 }  
 }  
  
 }  
  
 @Override  
 **public void** onChildChanged(DataSnapshot dataSnapshot, String s) {  
 String typingStatus = dataSnapshot.getValue().toString();  
 **if** (typingStatus.equals(**"Typing"**)) {  
 getSupportActionBar().setSubtitle(typingStatus + **"..."**);  
 } **else** {  
 *// check if online* getSupportActionBar().setSubtitle(**"Online"**);  
 }  
 }  
  
 @Override  
 **public void** onChildRemoved(DataSnapshot dataSnapshot) {  
 *//layout.removeAllViews();* **if** (dataSnapshot.getKey().equals(**"TypingStatus"**)) {  
 getSupportActionBar().setSubtitle(**"Online"**);  
  
 }  
 }  
  
 @Override  
 **public void** onChildMoved(DataSnapshot dataSnapshot, String s) {  
  
 }  
  
 @Override  
 **public void** onCancelled(FirebaseError firebaseError) {  
  
 }  
 };  
 **refFriendListener** = **new** ChildEventListener() {  
 @Override  
 **public void** onChildAdded(DataSnapshot dataSnapshot, String s) {  
 **if** (dataSnapshot.getKey().equals(**"Status"**)) {  
 *// check if subtitle is not Typing* CharSequence subTitle = getSupportActionBar().getSubtitle();  
 **if** (subTitle != **null**) {  
 **if** (!subTitle.equals(**"Typing..."**)) {  
 String friendStatus = dataSnapshot.getValue().toString();  
 **if** (!friendStatus.equals(**"Online"**)) {  
 friendStatus = Tools.*lastSeenProper*(friendStatus);  
 }  
 getSupportActionBar().setSubtitle(friendStatus);  
 }  
 } **else** {  
 String friendStatus = dataSnapshot.getValue().toString();  
 **if** (!friendStatus.equals(**"Online"**)) {  
 friendStatus = Tools.*lastSeenProper*(friendStatus);  
 }  
 getSupportActionBar().setSubtitle(friendStatus);  
 }  
  
  
 }  
  
 }  
  
 @Override  
 **public void** onChildChanged(DataSnapshot dataSnapshot, String s) {  
 String friendStatus = dataSnapshot.getValue().toString();  
 **if** (!friendStatus.equals(**"Online"**)) {  
 friendStatus = Tools.*lastSeenProper*(friendStatus);  
 }  
 getSupportActionBar().setSubtitle(friendStatus);  
 }  
  
 @Override  
 **public void** onChildRemoved(DataSnapshot dataSnapshot) {  
  
 }  
  
 @Override  
 **public void** onChildMoved(DataSnapshot dataSnapshot, String s) {  
  
 }  
  
 @Override  
 **public void** onCancelled(FirebaseError firebaseError) {  
  
 }  
 };  
 Bundle extras = getIntent().getExtras();  
 **friendEmail** = extras.getString(**"FriendEmail"**);  
 List<Message> chatList = **db**.getChat(**user**.**Email**, **friendEmail**, 1);  
 **for** (Message item : chatList) {  
 **int** messageType = item.**FromMail**.equals(**user**.**Email**) ? 1 : 2;  
 appendMessage(item.**Message**, item.**SentDate**, messageType, **false**);  
 }  
  
 **friendFullName** = extras.getString(**"FriendFullName"**);  
  
 getSupportActionBar().setTitle(**friendFullName**);  
 **reference1** = **new** Firebase(StaticInfo.*MessagesEndPoint* + **"/"** + **user**.**Email** + **"-@@-"** + **friendEmail**);  
 **reference2** = **new** Firebase(StaticInfo.*MessagesEndPoint* + **"/"** + **friendEmail** + **"-@@-"** + **user**.**Email**);  
 **refFriend** = **new** Firebase(StaticInfo.*UsersURL* + **"/"** + **friendEmail**);  
 **refNotMess** = **new** Firebase(StaticInfo.*NotificationEndPoint* + **"/"** + **friendEmail**);  
 **refFriend**.addChildEventListener(**refFriendListener**);  
 StaticInfo.*UserCurrentChatFriendEmail* = **friendEmail**;  
 **refUser** = **new** Firebase(StaticInfo.*UsersURL* + **"/"** + **user**.**Email**);  
 **submit\_btn** = (FloatingActionButton) findViewById(R.id.***submit\_btn***);  
  
 **messageArea**.addTextChangedListener(**new** TextWatcher() {  
 @Override  
 **public void** beforeTextChanged(CharSequence s, **int** start, **int** count, **int** after) {  
  
 }  
  
 @Override  
 **public void** onTextChanged(CharSequence s, **int** start, **int** before, **int** count) {  
 **if** (**messageArea**.getText().toString().length() == 0) {  
 **reference2**.child(StaticInfo.*TypingStatus*).setValue(**""**);  
 } **else if** (**messageArea**.getText().toString().length() == 1) {  
 **reference2**.child(StaticInfo.*TypingStatus*).setValue(**"Typing"**);  
 *// change color here  
 // submit\_btn.setColorFilter(R.color.colorPrimary);* }  
 }  
  
 @Override  
 **public void** afterTextChanged(Editable s) {  
  
 }  
 });  
  
 View rootView = findViewById(R.id.***rootLayout***);  
 EmojiconEditText emojiconEditText = (EmojiconEditText) findViewById(R.id.***et\_Message***);  
 ImageView emojiImageView = (ImageView) findViewById(R.id.***emoji\_btn***);  
  
 **final** EmojIconActions emojIcon = **new** EmojIconActions(**this**, rootView, emojiconEditText, emojiImageView, **"#1c2764"**, **"#e8e8e8"**, **"#f4f4f4"**);  
 emojIcon.ShowEmojIcon();  
  
 emojIcon.setKeyboardListener(**new** EmojIconActions.KeyboardListener() {  
 @Override  
 **public void** onKeyboardOpen() {  
  
 **scrollView**.post(**new** Runnable() {  
 @Override  
 **public void** run() {  
 **scrollView**.fullScroll(View.***FOCUS\_DOWN***);  
 }  
 });  
 }  
  
 @Override  
 **public void** onKeyboardClose() {  
  
 }  
 });  
  
 **final** SwipeRefreshLayout swipeRefreshLayout = (SwipeRefreshLayout) findViewById(R.id.***swiperefresh***);  
  
 swipeRefreshLayout.setOnRefreshListener(**new** SwipeRefreshLayout.OnRefreshListener() {  
 @Override  
 **public void** onRefresh() {  
 List<Message> chatList = **db**.getChat(**user**.**Email**, **friendEmail**, **pageNo**);  
 **layout**.removeAllViews();  
 **for** (Message item : chatList) {  
 **int** messageType = item.**FromMail**.equals(**user**.**Email**) ? 1 : 2;  
 appendMessage(item.**Message**, item.**SentDate**, messageType, **true**);  
 }  
 swipeRefreshLayout.setRefreshing(**false**);  
 **pageNo**++;  
 }  
 });  
   
   
 *// getSupportActionBar().setDisplayHomeAsUpEnabled(true);* toolbar.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent intent = **new** Intent(ActivityChat.**this**, ActivityFriendProfile.**class**);  
 intent.putExtra(**"Email"**, **friendEmail**);  
 startActivityForResult(intent, StaticInfo.*ChatAciviityRequestCode*);  
 }  
 });  
  
 }  
  
 @Override  
 **protected void** onStart() {  
 **super**.onStart();  
 Bundle extras = getIntent().getExtras();  
 **friendEmail** = extras.getString(**"FriendEmail"**);  
  
 *// getSupportActionBar().setTitle(extras.getString("FriendFullName"));  
 // getSupportActionBar().setIcon(R.drawable.dp\_placeholder\_sm);* **scrollView**.post(**new** Runnable() {  
 @Override  
 **public void** run() {  
 **scrollView**.fullScroll(View.***FOCUS\_DOWN***);  
 }  
 });  
 StaticInfo.*UserCurrentChatFriendEmail* = **friendEmail**;  
 *// update status to online* **refUser**.child(**"Status"**).setValue(**"Online"**);  
 **reference1**.addChildEventListener(**reference1Listener**);  
 }  
  
  
 @Override  
 **protected void** onPause() {  
 **super**.onPause();  
 **reference1**.removeEventListener(**reference1Listener**);  
 }  
  
 @Override  
 **protected void** onRestart() {  
 **super**.onRestart();  
 StaticInfo.*UserCurrentChatFriendEmail* = **friendEmail**;  
 **refUser**.child(**"Status"**).setValue(**"Online"**);  
 }  
  
 @Override  
 **protected void** onStop() {  
 **super**.onStop();  
 StaticInfo.*UserCurrentChatFriendEmail* = **""**;  
 **reference1**.removeEventListener(**reference1Listener**);  
 **reference2**.child(StaticInfo.*TypingStatus*).setValue(**""**);  
 }  
  
 @Override  
 **protected void** onDestroy() {  
 **super**.onDestroy();  
 StaticInfo.*UserCurrentChatFriendEmail* = **""**;  
 *// set last seen* DateFormat dateFormat = **new** SimpleDateFormat(**"dd MM yy hh:mm a"**);  
 Date date = **new** Date();  
 **refUser**.child(**"Status"**).setValue(dateFormat.format(date));  
 **reference1**.removeEventListener(**reference1Listener**);  
 **reference2**.child(StaticInfo.*TypingStatus*).setValue(**""**);  
 }  
  
 @Override  
 **protected void** onNewIntent(Intent intent) {  
 **super**.onNewIntent(intent);  
  
 Bundle extras = intent.getExtras();  
 **layout**.removeAllViews();  
 **friendEmail** = extras.getString(**"FriendEmail"**);  
 **friendFullName** = extras.getString(**"FriendFullName"**);  
 getSupportActionBar().setTitle(**friendFullName**);  
 List<Message> chatList = **db**.getChat(**user**.**Email**, **friendEmail**, 1);  
 **for** (Message item : chatList) {  
 **int** messageType = item.**FromMail**.equals(**user**.**Email**) ? 1 : 2;  
 appendMessage(item.**Message**, item.**SentDate**, messageType, **false**);  
 }  
  
 StaticInfo.*UserCurrentChatFriendEmail* = **friendEmail**;  
 **reference1**.removeEventListener(**reference1Listener**);  
 **reference1** = **new** Firebase(StaticInfo.*MessagesEndPoint* + **"/"** + **user**.**Email** + **"-@@-"** + **friendEmail**);  
 **reference1**.addChildEventListener(**reference1Listener**);  
  
 **refFriend**.removeEventListener(**refFriendListener**);  
 **refFriend** = **new** Firebase(StaticInfo.*UsersURL* + **"/"** + **friendEmail**);  
 **refFriend**.addChildEventListener(**refFriendListener**);  
  
 **reference2** = **new** Firebase(StaticInfo.*MessagesEndPoint* + **"/"** + **friendEmail** + **"-@@-"** + **user**.**Email**);  
  
 }  
  
 **public void** btn\_SendMessageClick(View view) {  
  
 String message = **messageArea**.getText().toString().trim();  
 **messageArea**.setText(**""**);  
 **if** (!message.equals(**""**)) {  
 Map<String, String> map = **new** HashMap<>();  
 map.put(**"Message"**, message);  
 map.put(**"SenderEmail"**, **user**.**Email**);  
 map.put(**"FirstName"**, **user**.**FirstName**);  
 map.put(**"LastName"**, **user**.**LastName**);  
  
 DateFormat dateFormat = **new** SimpleDateFormat(**"dd MM yy hh:mm a"**);  
 Date date = **new** Date();  
 String sentDate = dateFormat.format(date);  
  
 map.put(**"SentDate"**, sentDate);  
 *//reference1.push().setValue(map);* **reference2**.push().setValue(map);  
 **refNotMess**.push().setValue(map);  
  
 *// save in local db* **db**.saveMessageOnLocakDB(**user**.**Email**, **friendEmail**, message, sentDate);  
  
 *// appendmessage* appendMessage(message, sentDate, 1, **false**);  
  
 }  
 }  
  
 **public void** appendMessage(String mess, String sentDate, **int** messType, **final boolean** scrollUp) {  
  
 EmojiconTextView textView = **new** EmojiconTextView(**this**);  
 textView.setEmojiconSize(30);  
 sentDate = Tools.*messageSentDateProper*(sentDate);  
 SpannableString dateString = **new** SpannableString(sentDate);  
 dateString.setSpan(**new** RelativeSizeSpan(0.7f), 0, sentDate.length(), 0);  
 dateString.setSpan(**new** ForegroundColorSpan(Color.***GRAY***), 0, sentDate.length(), 0);  
  
 textView.setText(mess + **"\n"**);  
 textView.append(dateString);  
 textView.setTextColor(Color.*parseColor*(**"#000000"**));  
  
  
 LinearLayout.LayoutParams lp = **new** LinearLayout.LayoutParams(  
 ViewGroup.LayoutParams.***WRAP\_CONTENT***,  
 ViewGroup.LayoutParams.***WRAP\_CONTENT***,  
 6f  
 );  
 lp.setMargins(0, 0, 0, 5);  
 *// 1 user* **if** (messType == 1) {  
 textView.setBackgroundResource(R.drawable.***messagebg1***);  
 lp.**gravity** = Gravity.***RIGHT***;  
 }  
 *// 2 friend* **else** {  
 textView.setBackgroundResource(R.drawable.***messagebg2***);  
 lp.**gravity** = Gravity.***LEFT***;  
 }  
  
 textView.setPadding(12, 4, 12, 4);  
  
 textView.setLayoutParams(lp);  
 **layout**.addView(textView);  
 **scrollView**.post(**new** Runnable() {  
 @Override  
 **public void** run() {  
 **if** (scrollUp)  
 **scrollView**.fullScroll(View.***FOCUS\_UP***);  
 **else  
 scrollView**.fullScroll(View.***FOCUS\_DOWN***);  
 }  
 });  
 }  
  
 @Override  
 **public boolean** onCreateOptionsMenu(Menu menu) {  
 getMenuInflater().inflate(R.menu.***menu\_chat***, menu);  
 **return true**;  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **int** id = item.getItemId();  
 **if** (id == R.id.***menu\_deleteConservation***) {  
 **new** AlertDialog.Builder(**this**)  
 .setTitle(**friendFullName**)  
 .setMessage(**"Are you sure to delete this chat?"**)  
 .setPositiveButton(**"Delete"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 **db**.deleteChat(**user**.**Email**, **friendEmail**);  
 **layout**.removeAllViews();  
 }  
 })  
 .setNegativeButton(android.R.string.***no***, **null**)  
 .show();  
 **return true**;  
 }  
 **if** (id == R.id.***menu\_deleteContact***) {  
 **new** AlertDialog.Builder(**this**)  
 .setTitle(**friendFullName**)  
 .setMessage(**"Are you sure to delete this contact?"**)  
 .setPositiveButton(**"Delete"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 Firebase ref = **new** Firebase(StaticInfo.*EndPoint* + **"/friends/"** + **user**.**Email** + **"/"** + **friendEmail**);  
 ref.removeValue();  
 *// delete from local database* **db**.deleteFriendByEmailFromLocalDB(**friendEmail**);  
 finish();  
 }  
 })  
 .setNegativeButton(android.R.string.***no***, **null**)  
 .show();  
 **return true**;  
 }  
  
 **if** (id == R.id.***menu\_friendProfile***) {  
 Intent intent = **new** Intent(ActivityChat.**this**, ActivityFriendProfile.**class**);  
 intent.putExtra(**"Email"**, **friendEmail**);  
 startActivityForResult(intent, StaticInfo.*ChatAciviityRequestCode*);  
 }  
  
 **if** (id == R.id.***menu\_phonecall***) {  
 Intent intentp = **new** Intent(ActivityChat.**this**, phonecall.**class**);  
 intentp.putExtra(**"Email"**, **friendEmail**);  
 startActivityForResult(intentp, StaticInfo.*ChatAciviityRequestCode*);  
 }  
 **return true**;  
 }  
  
 @Override  
 **protected void** onActivityResult(**int** requestCode, **int** resultCode, Intent data) {  
  
 **if** (requestCode == StaticInfo.*ChatAciviityRequestCode* && resultCode == Activity.***RESULT\_OK***) {  
 User updatedFriend = **db**.getFriendByEmailFromLocalDB(**friendEmail**);  
 **friendFullName** = updatedFriend.**FirstName**;  
 getSupportActionBar().setTitle(updatedFriend.**FirstName**);  
 }  
  
 **super**.onActivityResult(requestCode, resultCode, data);  
 }  
  
  
}

ActivityFriendProfile

**package** com.mys3soft.mys3chat;  
  
**import** android.app.Activity;  
**import** android.app.ProgressDialog;  
**import** android.content.DialogInterface;  
**import** android.content.Intent;  
**import** android.os.AsyncTask;  
**import** android.support.v7.app.AlertDialog;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.text.InputType;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.ImageButton;  
**import** android.widget.ListAdapter;  
**import** android.widget.ProgressBar;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** com.firebase.client.Firebase;  
**import** com.mys3soft.mys3chat.Models.StaticInfo;  
**import** com.mys3soft.mys3chat.Models.User;  
**import** com.mys3soft.mys3chat.Services.DataContext;  
**import** com.mys3soft.mys3chat.Services.IFireBaseAPI;  
**import** com.mys3soft.mys3chat.Services.LocalUserService;  
**import** com.mys3soft.mys3chat.Services.Tools;  
  
**import** org.json.JSONException;  
**import** org.json.JSONObject;  
**import** org.w3c.dom.Text;  
  
**import** java.io.IOException;  
**import** java.util.ArrayList;  
**import** java.util.HashMap;  
**import** java.util.Iterator;  
**import** java.util.List;  
**import** java.util.Map;  
  
**import** retrofit2.Call;  
  
  
**public class** ActivityFriendProfile **extends** AppCompatActivity {  
  
 String **friendEmail**;  
 TextView **tv\_FriendFullName**;  
 User **user**, **f**;  
 ProgressDialog **pd**;  
 Button **btn\_AddFriend**;  
 DataContext **db**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_friend\_profile***);  
 Firebase.*setAndroidContext*(**this**);  
 **btn\_AddFriend** = (Button) findViewById(R.id.***btn\_AddFriend***);  
 **pd** = **new** ProgressDialog(**this**);  
  
 **pd**.setMessage(**"Loading..."**);  
 **f** = **new** User();  
 Bundle extras = getIntent().getExtras();  
 **friendEmail** = extras.getString(**"Email"**);  
 **tv\_FriendFullName** = (TextView) findViewById(R.id.***tv\_FriendFullName\_L\_FriendProfile***);  
  
 **user** = LocalUserService.*getLocalUserFromPreferences*(**this**);  
  
 **db** = **new** DataContext(**this**, **null**, **null**, 1);  
  
 *// check if already friends otherwise get info from server* User friend = **db**.getFriendByEmailFromLocalDB(**friendEmail**);  
 **if** (friend.**Email** == **null**) {  
 ImageButton btn\_EditName = (ImageButton) findViewById(R.id.***btn\_EditName***);  
 btn\_EditName.setVisibility(View.***INVISIBLE***);  
 FindFriendsTask t = **new** FindFriendsTask();  
 t.execute();  
 } **else** {  
 **tv\_FriendFullName**.setText(Tools.*toProperName*(friend.**FirstName**) + **" "** + Tools.*toProperName*(friend.**LastName**));  
 **btn\_AddFriend**.setEnabled(**false**);  
 **btn\_AddFriend**.setText(**"Connected"**);  
  
  
 }  
  
  
 }  
  
  
 **public class** FindFriendsTask **extends** AsyncTask<Void, Void, String> {  
  
 @Override  
 **protected void** onPreExecute() {  
 **pd**.show();  
 }  
  
 @Override  
 **protected** String doInBackground(Void... params) {  
  
 IFireBaseAPI api = Tools.*makeRetroFitApi*();  
 Call<String> call = api.getAllUsersAsJsonString();  
  
 **try** {  
 **return** call.execute().body();  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 **pd**.hide();  
 }  
  
 **return null**;  
 }  
  
 @Override  
 **protected void** onPostExecute(String jsonListString) {  
 **try** {  
 JSONObject jsonObjectList = **new** JSONObject(jsonListString);  
 JSONObject item = jsonObjectList.getJSONObject(**friendEmail**);  
 **f**.**Email** = item.getString(**"Email"**);  
 **f**.**FirstName** = item.getString(**"FirstName"**);  
 **f**.**LastName** = item.getString(**"LastName"**);  
 **tv\_FriendFullName**.setText(Tools.*toProperName*(**f**.**FirstName**) + **" "** + Tools.*toProperName*(**f**.**LastName**));  
 **pd**.hide();  
 } **catch** (JSONException e1) {  
 e1.printStackTrace();  
 **pd**.hide();  
 }  
 }  
 }  
  
 **public void** btn\_SendFriendRequestClick(View view) {  
 Firebase firebase = **new** Firebase(**"https://mys3chat.firebaseio.com/friendrequests"**);  
 Firebase notifRef = **new** Firebase(StaticInfo.*NotificationEndPoint* + **"/"** + **friendEmail**);  
 Map<String, String> map = **new** HashMap<>();  
 map.put(**"FirstName"**, **user**.**FirstName**);  
 map.put(**"LastName"**, **user**.**LastName**);  
 firebase.child(Tools.*encodeString*(**f**.**Email**)).child(Tools.*encodeString*(**user**.**Email**)).setValue(map);  
 **btn\_AddFriend**.setEnabled(**false**);  
 **btn\_AddFriend**.setText(**"Request Sent"**);  
  
 map.put(**"SenderEmail"**, **user**.**Email**);  
 map.put(**"Message"**, **"Pending contact request"**);  
 map.put(**"NotificationType"**, **"2"**);  
  
 notifRef.push().setValue(map);  
  
 }  
  
 **public void** btn\_EditNameClick(View view) {  
  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(**this**);  
 builder.setTitle(**"Set display name"**);  
 **final** EditText et = **new** EditText(**this**);  
 et.setText(**tv\_FriendFullName**.getText());  
 et.setInputType(InputType.***TYPE\_TEXT\_FLAG\_CAP\_SENTENCES***);  
 builder.setView(et);  
 builder.setPositiveButton(**"Save"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 String newName = et.getText().toString();  
 **db**.setPreferedDisplayName(**friendEmail**, newName);  
 **tv\_FriendFullName**.setText(newName);  
 setResult(Activity.***RESULT\_OK***);  
 }  
 });  
 builder.setNegativeButton(**"Cancel"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 dialog.cancel();  
 }  
 });  
  
 builder.show();  
  
 }  
  
}

ActivityNotifications

**package** com.mys3soft.mys3chat;  
  
**import** android.os.AsyncTask;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.ListAdapter;  
**import** android.widget.ListView;  
**import** android.widget.ProgressBar;  
  
**import** com.firebase.client.ChildEventListener;  
**import** com.firebase.client.DataSnapshot;  
**import** com.firebase.client.Firebase;  
**import** com.firebase.client.FirebaseError;  
**import** com.mys3soft.mys3chat.Models.NotificationModel;  
**import** com.mys3soft.mys3chat.Models.StaticInfo;  
**import** com.mys3soft.mys3chat.Models.User;  
**import** com.mys3soft.mys3chat.Services.DataContext;  
**import** com.mys3soft.mys3chat.Services.IFireBaseAPI;  
**import** com.mys3soft.mys3chat.Services.LocalUserService;  
**import** com.mys3soft.mys3chat.Services.Tools;  
  
**import** org.json.JSONException;  
**import** org.json.JSONObject;  
  
**import** java.io.IOException;  
**import** java.util.ArrayList;  
**import** java.util.Iterator;  
**import** java.util.List;  
**import** java.util.Map;  
  
**import** retrofit2.Call;  
  
**public class** ActivityNotifications **extends** AppCompatActivity {  
  
 ListView **lv\_NotificationList**;  
 User **user**;  
 List<NotificationModel> **notificationList**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_notifications***);  
 Firebase.*setAndroidContext*(**this**);  
 }  
  
 @Override  
 **protected void** onStart() {  
 **super**.onStart();  
  
 **lv\_NotificationList** = (ListView) findViewById(R.id.***lv\_NoticicationList***);  
 **notificationList** = **new** ArrayList<>();  
 **user** = LocalUserService.*getLocalUserFromPreferences*(**this**);  
 Firebase reqRef = **new** Firebase(StaticInfo.*EndPoint* + **"/friendrequests/"** + **user**.**Email**);  
 reqRef.addChildEventListener(  
 **new** ChildEventListener() {  
 @Override  
 **public void** onChildAdded(DataSnapshot dataSnapshot, String s) {  
 Map map = dataSnapshot.getValue(Map.**class**);  
 String firstName = map.get(**"FirstName"**).toString();  
 String lastName = map.get(**"LastName"**).toString();  
 **final** String key = dataSnapshot.getKey();  
 NotificationModel not = **new** NotificationModel();  
 not.**FirstName** = firstName;  
 not.**LastName** = lastName;  
 not.**NotificationType** = 1; *// friend request* **notificationList**.add(not);  
 not.**EmailFrom** = key;  
 not.**FriendRequestFireBaseKey** = dataSnapshot.getKey();  
 not.**NotificationMessage** = Tools.*toProperName*(firstName) + **" "** + Tools.*toProperName*(lastName);  
 ListAdapter adp = **new** NotficationListAdapter(ActivityNotifications.**this**, **notificationList**);  
 **lv\_NotificationList**.setAdapter(adp);  
 }  
  
 @Override  
 **public void** onChildChanged(DataSnapshot dataSnapshot, String s) {  
  
 }  
  
 @Override  
 **public void** onChildRemoved(DataSnapshot dataSnapshot) {  
 String friendEmail = dataSnapshot.getKey();  
 **int** index = -1;  
 **for** (**int** i = 0; i < **notificationList**.size(); i++) {  
 NotificationModel item = **notificationList**.get(i);  
 **if** (item.**EmailFrom**.equals(friendEmail))  
 index = i;  
 }  
 **notificationList**.remove(index);  
 ListAdapter adp = **new** NotficationListAdapter(ActivityNotifications.**this**, **notificationList**);  
 **lv\_NotificationList**.setAdapter(adp);  
  
 }  
  
 @Override  
 **public void** onChildMoved(DataSnapshot dataSnapshot, String s) {  
  
 }  
  
 @Override  
 **public void** onCancelled(FirebaseError firebaseError) {  
  
 }  
 }  
 );  
  
  
 }  
  
}

ActivityProfile

**package** com.mys3soft.mys3chat;  
  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.widget.TextView;  
  
**import** com.mys3soft.mys3chat.Models.User;  
**import** com.mys3soft.mys3chat.Services.DataContext;  
**import** com.mys3soft.mys3chat.Services.LocalUserService;  
**import** com.mys3soft.mys3chat.Services.Tools;  
  
**import** org.w3c.dom.Text;  
  
**public class** ActivityProfile **extends** AppCompatActivity {  
  
 DataContext **db** = **new** DataContext(**this**, **null**, **null**, 1);  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_profile***);  
 User user = LocalUserService.*getLocalUserFromPreferences*(**this**);  
 TextView tv\_UserFullName = (TextView) findViewById(R.id.***tv\_UserFullName***);  
 tv\_UserFullName.setText(Tools.*toProperName*(user.**FirstName**) + **" "** + Tools.*toProperName*(user.**LastName**));  
 }  
}

ActivityLastchat

**package** com.mys3soft.mys3chat;  
  
**import** android.content.Context;  
**import** android.support.annotation.NonNull;  
**import** android.support.annotation.Nullable;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.TextView;  
  
**import** com.mys3soft.mys3chat.Models.Message;  
**import** com.mys3soft.mys3chat.Models.User;  
**import** com.mys3soft.mys3chat.Services.Tools;  
  
**import** java.util.List;  
  
  
  
**public class** AdapterLastChat **extends** ArrayAdapter<Message> {  
  
 **public** AdapterLastChat(@NonNull Context context, List<Message> messageList) {  
 **super**(context, R.layout.***custom\_lastchat\_row***, messageList);  
 }  
  
  
 @NonNull  
 @Override  
 **public** View getView(**int** position, @Nullable View convertView, @NonNull ViewGroup parent) {  
 LayoutInflater inflater = LayoutInflater.*from*(getContext());  
 View customView = inflater.inflate(R.layout.***custom\_lastchat\_row***, parent, **false**);  
 Message message = getItem(position);  
 TextView hiddenEmail = (TextView) customView.findViewById(R.id.***tv\_lastChat\_HiddenEmail***);  
 TextView tv\_Name = (TextView) customView.findViewById(R.id.***tv\_lastChat\_FriendFullName***);  
 TextView tv\_MessageDate = (TextView) customView.findViewById(R.id.***tv\_lastChat\_MessageDate***);  
 TextView tv\_Message = (TextView) customView.findViewById(R.id.***tv\_lastChat\_Message***);  
 hiddenEmail.setText(String.*valueOf*(message.**FromMail**));  
 tv\_Name.setText(message.**FriendFullName**);  
 String properDate = Tools.*messageSentDateProper*(message.**SentDate**);  
 tv\_MessageDate.setText(properDate);  
 **if** (message.**Message**.length() > 20){  
 message.**Message** = message.**Message**.substring(0,20);  
 }  
 tv\_Message.setText(message.**Message**);  
 **return** customView;  
 }  
  
}

Appreciver

**package** com.mys3soft.mys3chat;  
  
**import** android.content.BroadcastReceiver;  
**import** android.content.Context;  
**import** android.content.Intent;  
  
**import** com.mys3soft.mys3chat.Services.LocalUserService;  
  
**public class** AppReceiver **extends** BroadcastReceiver {  
  
 @Override  
 **public void** onReceive(Context context, Intent intent) {  
 context.startService(**new** Intent(context,AppService.**class**));  
 }  
}

Appservice

**package** com.mys3soft.mys3chat;  
  
**import** android.app.Notification;  
**import** android.app.NotificationManager;  
**import** android.app.PendingIntent;  
**import** android.app.Service;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.os.IBinder;  
**import** android.os.Vibrator;  
**import** android.provider.ContactsContract;  
**import** android.support.v7.app.NotificationCompat;  
  
**import** com.firebase.client.ChildEventListener;  
**import** com.firebase.client.DataSnapshot;  
**import** com.firebase.client.Firebase;  
**import** com.firebase.client.FirebaseError;  
**import** com.mys3soft.mys3chat.Models.StaticInfo;  
**import** com.mys3soft.mys3chat.Models.User;  
**import** com.mys3soft.mys3chat.Services.DataContext;  
**import** com.mys3soft.mys3chat.Services.LocalUserService;  
**import** com.mys3soft.mys3chat.Services.Tools;  
  
**import** java.util.Map;  
  
**public class** AppService **extends** Service {  
 **public** AppService() {  
 }  
  
 @Override  
 **public** IBinder onBind(Intent intent) {  
 **return null**;  
 }  
  
 @Override  
 **public int** onStartCommand(Intent intent, **int** flags, **int** startId) {  
 Firebase.*setAndroidContext*(getApplicationContext());  
 User user = LocalUserService.*getLocalUserFromPreferences*(getApplicationContext());  
 **final** Firebase reference = **new** Firebase(StaticInfo.*NotificationEndPoint* + **"/"** + user.**Email**);  
 reference.addChildEventListener(  
 **new** ChildEventListener() {  
 @Override  
 **public void** onChildAdded(DataSnapshot dataSnapshot, String s) {  
 **if** (LocalUserService.*getLocalUserFromPreferences*(getApplicationContext()).**Email** != **null**) {  
 Map map = dataSnapshot.getValue(Map.**class**);  
 String mess = map.get(**"Message"**).toString();  
 String senderEmail = map.get(**"SenderEmail"**).toString();  
 String senderFullName = Tools.*toProperName*(map.get(**"FirstName"**).toString()) + **" "** + Tools.*toProperName*(  
 map.get(**"LastName"**).toString());  
 **int** notificationType = 1; *// Message* notificationType = map.get(**"NotificationType"**) == **null** ? 1 : Integer.*parseInt*(map.get(**"NotificationType"**).toString());  
 *// check if user is on chat activity with senderEmail* **if** (!StaticInfo.*UserCurrentChatFriendEmail*.equals(senderEmail)) {  
 notifyUser(senderEmail, senderFullName, mess, notificationType);  
 *// remove notification* reference.child(dataSnapshot.getKey()).removeValue();  
 } **else** {  
 reference.child(dataSnapshot.getKey()).removeValue();  
 }  
 }  
 }  
  
 @Override  
 **public void** onChildChanged(DataSnapshot dataSnapshot, String s) {  
  
 }  
  
 @Override  
 **public void** onChildRemoved(DataSnapshot dataSnapshot) {  
  
 }  
  
 @Override  
 **public void** onChildMoved(DataSnapshot dataSnapshot, String s) {  
  
 }  
  
 @Override  
 **public void** onCancelled(FirebaseError firebaseError) {  
  
 }  
 }  
 );  
 **return *START\_STICKY***;  
 }  
  
 @Override  
 **public void** onDestroy() {  
 **super**.onDestroy();  
 *// check if user is login* **if** (LocalUserService.*getLocalUserFromPreferences*(getApplicationContext()).**Email** != **null**) {  
 sendBroadcast(**new** Intent(**"com.mys3soft.mys3chat.restartservice"**));  
 }  
  
  
 }  
  
 **private void** notifyUser(String friendEmail, String senderFullName, String mess, **int** notificationType) {  
 NotificationCompat.Builder not = **new** NotificationCompat.Builder(getApplicationContext());  
 not.setAutoCancel(**true**);  
 not.setSmallIcon(R.mipmap.***ic\_launcher\_round***);  
 not.setTicker(**"New Message"**);  
 not.setWhen(System.*currentTimeMillis*());  
 not.setContentText(mess);  
 Intent i;  
 *// 1) Message 3) Contact Request Accepted* **if** (notificationType == 1 || notificationType == 3) {  
 i = **new** Intent(getApplicationContext(), ActivityChat.**class**);  
 DataContext db = **new** DataContext(getApplicationContext(), **null**, **null**, 1);  
 User frnd = db.getFriendByEmailFromLocalDB(friendEmail);  
 **if** (frnd.**FirstName** != **null**) {  
 not.setContentTitle(frnd.**FirstName** + **" "** + frnd.**LastName**);  
 i.putExtra(**"FriendFullName"**, frnd.**FirstName** + **" "** + frnd.**LastName**);  
 } **else** {  
 not.setContentTitle(senderFullName);  
 i.putExtra(**"FriendFullName"**, senderFullName);  
 }  
 }  
 *// Contact Request* **else if** (notificationType == 2) {  
 i = **new** Intent(getApplicationContext(), ActivityNotifications.**class**);  
 not.setContentTitle(senderFullName);  
 } **else** {  
 i = **null**;  
 }  
 i.putExtra(**"FriendEmail"**, friendEmail);  
 **int** uniqueID = Tools.*createUniqueIdPerUser*(friendEmail);  
 PendingIntent pendingIntent = PendingIntent.*getActivity*(getApplicationContext(), uniqueID, i, PendingIntent.***FLAG\_UPDATE\_CURRENT***);  
 not.setContentIntent(pendingIntent);  
 NotificationManager nm = (NotificationManager) getSystemService(***NOTIFICATION\_SERVICE***);  
 not.setDefaults(Notification.***DEFAULT\_ALL***);  
 nm.notify(uniqueID, not.build());  
  
 }  
}

FriendListAdapter

**package** com.mys3soft.mys3chat;  
  
  
**import** android.content.Context;  
**import** android.support.annotation.NonNull;  
**import** android.support.annotation.Nullable;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.TextView;  
  
**import** com.mys3soft.mys3chat.Models.User;  
**import** com.mys3soft.mys3chat.Services.Tools;  
  
**import** java.util.List;  
  
**public class** FriendListAdapter **extends** ArrayAdapter<User> {  
  
  
 **public** FriendListAdapter(@NonNull Context context, List<User> contactList) {  
 **super**(context, R.layout.***custom\_friend\_list\_row***, contactList);  
 }  
  
 @NonNull  
 @Override  
 **public** View getView(**int** position, @Nullable View convertView, @NonNull ViewGroup parent) {  
 LayoutInflater inflater = LayoutInflater.*from*(getContext());  
 View customView = inflater.inflate(R.layout.***custom\_friend\_list\_row***, parent, **false**);  
 User user = getItem(position);  
 TextView hiddenEmail = (TextView) customView.findViewById(R.id.***tv\_HiddenEmail***);  
 TextView tv\_Name = (TextView) customView.findViewById(R.id.***tv\_FriendFullName***);  
 hiddenEmail.setText(String.*valueOf*(user.**Email**));  
 tv\_Name.setText(Tools.*toProperName*(user.**FirstName**) + **" "** + Tools.*toProperName*(user.**LastName**));  
 **return** customView;  
 }  
  
}

NotificationListAdapter

**package** com.mys3soft.mys3chat;  
  
  
**import** android.content.Context;  
**import** android.content.DialogInterface;  
**import** android.graphics.Color;  
**import** android.graphics.drawable.Drawable;  
**import** android.support.annotation.LayoutRes;  
**import** android.support.annotation.NonNull;  
**import** android.support.annotation.Nullable;  
**import** android.support.annotation.StringDef;  
**import** android.support.v4.app.ListFragment;  
**import** android.support.v7.app.AlertDialog;  
**import** android.text.Layout;  
**import** android.view.Gravity;  
**import** android.view.LayoutInflater;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.Button;  
**import** android.widget.ImageButton;  
**import** android.widget.LinearLayout;  
**import** android.widget.RelativeLayout;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**import** com.firebase.client.Firebase;  
**import** com.mys3soft.mys3chat.Models.NotificationModel;  
**import** com.mys3soft.mys3chat.Models.StaticInfo;  
**import** com.mys3soft.mys3chat.Models.User;  
**import** com.mys3soft.mys3chat.Services.DataContext;  
**import** com.mys3soft.mys3chat.Services.LocalUserService;  
  
**import** java.util.HashMap;  
**import** java.util.List;  
**import** java.util.Map;  
  
**public class** NotficationListAdapter **extends** ArrayAdapter<NotificationModel> {  
  
 **private** Context **con**;  
 **private** ImageButton **acceptBtn**;  
 **private** ImageButton **rejectBtn**;  
  
  
 **public** NotficationListAdapter(@NonNull Context context, List<NotificationModel> list) {  
 **super**(context, R.layout.***custom\_notication\_row***, list);  
 **con** = context;  
  
 }  
  
 @NonNull  
 @Override  
 **public** View getView(**int** position, @Nullable View convertView, @NonNull ViewGroup parent) {  
  
 LayoutInflater inflater = LayoutInflater.*from*(getContext());  
 View customView = inflater.inflate(R.layout.***custom\_notication\_row***, parent, **false**);  
 NotificationModel model = getItem(position);  
 *// get layout* LinearLayout layout = (LinearLayout) customView.findViewById(R.id.***layout\_CustomNotificationRow***);  
  
 *// make components according to model and append to layout* TextView tv\_NotficationMessage = (TextView) customView.findViewById(R.id.***tv\_NotificationMessage***);  
 tv\_NotficationMessage.setText(model.**NotificationMessage**);  
  
 *// friend request* **if** (model.**NotificationType** == 1) {  
 *// make button and append  
// acceptBtn = new Button(getContext());  
// rejectBtn = new Button(getContext());* **acceptBtn** = **new** ImageButton(getContext());  
 **rejectBtn** = **new** ImageButton(getContext());  
  
 **acceptBtn**.setBackgroundColor(Color.***TRANSPARENT***);  
 **rejectBtn**.setBackgroundColor(Color.***TRANSPARENT***);  
  
 **acceptBtn**.setImageResource(R.drawable.***emoji\_2705***);  
 **rejectBtn**.setImageResource(R.drawable.***emoji\_274c***);  
  
 setCustomOnClick(**acceptBtn**, model.**EmailFrom**, model.**FirstName**, model.**LastName**);  
 onRejectClick(**rejectBtn**, position, model.**FirstName** + **" "** + model.**LastName**);  
 *// set layout params* LinearLayout.LayoutParams layoutParams = **new** LinearLayout.LayoutParams(  
 ViewGroup.LayoutParams.***WRAP\_CONTENT***,  
 ViewGroup.LayoutParams.***WRAP\_CONTENT*** );  
 layoutParams.**gravity** = Gravity.***CENTER***;  
  
 **acceptBtn**.setLayoutParams(layoutParams);  
 **rejectBtn**.setLayoutParams(layoutParams);  
 **acceptBtn**.setPadding(4, 4, 4, 4);  
 **rejectBtn**.setPadding(4, 4, 4, 4);  
 layout.addView(**acceptBtn**);  
 layout.addView(**rejectBtn**);  
 }  
 **return** customView;  
 }  
  
  
 **private void** setCustomOnClick(**final** ImageButton btn, **final** String friendEmail, **final** String friendFirstName, **final** String friendLastName) {  
  
 btn.setOnClickListener(  
 **new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 User user = LocalUserService.*getLocalUserFromPreferences*(**con**);  
 *// add to friends and remove from requests* Firebase fireBase = **new** Firebase(StaticInfo.*FriendsURL*);  
 *// set each other friends* Map<String, String> map1 = **new** HashMap<>();  
 map1.put(**"Email"**, friendEmail);  
 map1.put(**"FirstName"**, friendFirstName);  
 map1.put(**"LastName"**, friendLastName);  
 fireBase.child(user.**Email**).child(friendEmail).setValue(map1);  
  
 Map<String, String> map2 = **new** HashMap<>();  
 map2.put(**"Email"**, user.**Email**);  
 map2.put(**"FirstName"**, user.**FirstName**);  
 map2.put(**"LastName"**, user.**LastName**);  
 fireBase.child(friendEmail).child(user.**Email**).setValue(map2);  
  
 Firebase frRequ = **new** Firebase(StaticInfo.*EndPoint* + **"/friendrequests"**);  
 frRequ.child(user.**Email**).child(friendEmail).removeValue();  
 **acceptBtn**.setEnabled(**false**);  
  
 Toast.*makeText*(**con**, **"Accepted"**, Toast.***LENGTH\_SHORT***).show();  
 **rejectBtn**.setEnabled(**false**);  
  
 Map<String, String> notMap = **new** HashMap<String, String>();  
 notMap.put(**"SenderEmail"**, user.**Email**);  
 notMap.put(**"FirstName"**, user.**FirstName**);  
 notMap.put(**"LastName"**, user.**LastName**);  
 notMap.put(**"Message"**, **"Contact request accepted start chating... "**);  
 *// accepted contact reques* notMap.put(**"NotificationType"**, **"3"**);  
 Firebase notRef = **new** Firebase(StaticInfo.*NotificationEndPoint* + **"/"** + friendEmail);  
 notRef.push().setValue(notMap);  
 }  
 }  
 );  
  
  
 }  
  
 **private void** onRejectClick(**final** ImageButton btn, **final int** modelPosition, **final** String friedFullName) {  
 btn.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **new** AlertDialog.Builder(**con**)  
 .setTitle(friedFullName)  
 .setMessage(**"Are you sure to reject this contact request?"**)  
 .setPositiveButton(**"Reject"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 User user = LocalUserService.*getLocalUserFromPreferences*(**con**);  
 Firebase fireBase = **new** Firebase(StaticInfo.*FriendRequestsEndPoint* + **"/"** + user.**Email** + **"/"** + getItem(modelPosition).**FriendRequestFireBaseKey**);  
 fireBase.removeValue();  
 **rejectBtn**.setEnabled(**false**);  
 **acceptBtn**.setEnabled(**false**);  
 Toast.*makeText*(**con**, **"Rejected"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 })  
 .setNegativeButton(android.R.string.***no***, **null**)  
 .show();  
 }  
 });  
  
 }  
}

Phonecall

**package** com.mys3soft.mys3chat;  
  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
  
**public class** phonecall **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_phonecall***);  
 }  
}

callActivity

**package** com.quickblox.sample.videochat.java.activities;  
  
**import** android.annotation.TargetApi;  
**import** android.app.Activity;  
**import** android.app.PendingIntent;  
**import** android.content.ComponentName;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.content.ServiceConnection;  
**import** android.content.SharedPreferences;  
**import** android.net.Uri;  
**import** android.os.Build;  
**import** android.os.Bundle;  
**import** android.os.Handler;  
**import** android.os.IBinder;  
**import** android.os.Looper;  
**import** android.os.SystemClock;  
**import** android.preference.PreferenceManager;  
**import** android.provider.Settings;  
**import** android.util.Log;  
**import** android.view.View;  
**import** android.view.ViewGroup;  
**import** android.widget.LinearLayout;  
**import** android.widget.TextView;  
  
**import** com.quickblox.chat.QBChatService;  
**import** com.quickblox.sample.videochat.java.R;  
**import** com.quickblox.sample.videochat.java.db.QbUsersDbManager;  
**import** com.quickblox.sample.videochat.java.fragments.AudioConversationFragment;  
**import** com.quickblox.sample.videochat.java.fragments.BaseConversationFragment;  
**import** com.quickblox.sample.videochat.java.fragments.ConversationFragmentCallback;  
**import** com.quickblox.sample.videochat.java.fragments.IncomeCallFragment;  
**import** com.quickblox.sample.videochat.java.fragments.IncomeCallFragmentCallbackListener;  
**import** com.quickblox.sample.videochat.java.fragments.ScreenShareFragment;  
**import** com.quickblox.sample.videochat.java.fragments.VideoConversationFragment;  
**import** com.quickblox.sample.videochat.java.services.CallService;  
**import** com.quickblox.sample.videochat.java.services.LoginService;  
**import** com.quickblox.sample.videochat.java.utils.Consts;  
**import** com.quickblox.sample.videochat.java.utils.ErrorUtils;  
**import** com.quickblox.sample.videochat.java.utils.FragmentExecuotr;  
**import** com.quickblox.sample.videochat.java.utils.PermissionsChecker;  
**import** com.quickblox.sample.videochat.java.utils.QBEntityCallbackImpl;  
**import** com.quickblox.sample.videochat.java.utils.SettingsUtil;  
**import** com.quickblox.sample.videochat.java.utils.SharedPrefsHelper;  
**import** com.quickblox.sample.videochat.java.utils.ToastUtils;  
**import** com.quickblox.sample.videochat.java.utils.UsersUtils;  
**import** com.quickblox.sample.videochat.java.utils.WebRtcSessionManager;  
**import** com.quickblox.users.model.QBUser;  
**import** com.quickblox.videochat.webrtc.AppRTCAudioManager;  
**import** com.quickblox.videochat.webrtc.BaseSession;  
**import** com.quickblox.videochat.webrtc.QBRTCScreenCapturer;  
**import** com.quickblox.videochat.webrtc.QBRTCSession;  
**import** com.quickblox.videochat.webrtc.QBRTCTypes;  
**import** com.quickblox.videochat.webrtc.callbacks.QBRTCClientSessionCallbacks;  
**import** com.quickblox.videochat.webrtc.callbacks.QBRTCClientVideoTracksCallbacks;  
**import** com.quickblox.videochat.webrtc.callbacks.QBRTCSessionEventsCallback;  
**import** com.quickblox.videochat.webrtc.callbacks.QBRTCSessionStateCallback;  
**import** com.quickblox.videochat.webrtc.view.QBRTCVideoTrack;  
  
**import** org.jivesoftware.smack.AbstractConnectionListener;  
**import** org.jivesoftware.smack.ConnectionListener;  
**import** org.webrtc.CameraVideoCapturer;  
  
**import** java.util.ArrayList;  
**import** java.util.HashMap;  
**import** java.util.List;  
**import** java.util.Map;  
  
**import** androidx.fragment.app.Fragment;  
  
*/\*\*  
 \* QuickBlox team  
 \*/***public class** CallActivity **extends** BaseActivity **implements** IncomeCallFragmentCallbackListener,  
 QBRTCSessionStateCallback<QBRTCSession>, QBRTCClientSessionCallbacks,  
 ConversationFragmentCallback, ScreenShareFragment.OnSharingEvents {  
 **private static final** String ***TAG*** = CallActivity.**class**.getSimpleName();  
  
 **public static final** String ***INCOME\_CALL\_FRAGMENT*** = **"income\_call\_fragment"**;  
 **public static final int *REQUEST\_PERMISSION\_SETTING*** = 545;  
  
 **private** ArrayList<CurrentCallStateCallback> **currentCallStateCallbackList** = **new** ArrayList<>();  
 **private** QbUsersDbManager **dbManager** = QbUsersDbManager.*getInstance*(**this**);  
 **private** Handler **showIncomingCallWindowTaskHandler**;  
 **private** ConnectionListenerImpl **connectionListener**;  
 **private** ServiceConnection **callServiceConnection**;  
 **private** Runnable **showIncomingCallWindowTask**;  
 **private boolean isInComingCall** = **false**;  
 **private** List<Integer> **opponentsIdsList**;  
 **private** SharedPreferences **sharedPref**;  
 **private boolean isVideoCall** = **false**;  
 **private** PermissionsChecker **checker**;  
 **private** CallService **callService**;  
  
 **public static void** start(Context context, **boolean** isIncomingCall) {  
 Intent intent = **new** Intent(context, CallActivity.**class**);  
 intent.addFlags(Intent.***FLAG\_ACTIVITY\_NEW\_TASK***);  
 intent.putExtra(Consts.***EXTRA\_IS\_INCOMING\_CALL***, isIncomingCall);  
 SharedPrefsHelper.*getInstance*().save(Consts.***EXTRA\_IS\_INCOMING\_CALL***, isIncomingCall);  
 context.startActivity(intent);  
 CallService.*start*(context);  
 }  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 **checker** = **new** PermissionsChecker(**this**);  
 }  
  
 **private void** initScreen() {  
 **callService**.setCallTimerCallback(**new** CallTimerCallback());  
 **isVideoCall** = **callService**.isVideoCall();  
  
 **opponentsIdsList** = **callService**.getOpponents();  
 PreferenceManager.*setDefaultValues*(**this**, R.xml.***preferences***, **false**);  
 **sharedPref** = PreferenceManager.*getDefaultSharedPreferences*(**this**);  
  
 initSettingsStrategy();  
 addListeners();  
  
 **if** (**callService**.isCallMode()) {  
 checkPermission();  
 **if** (**callService**.isSharingScreenState()) {  
 startScreenSharing(**null**);  
 **return**;  
 }  
 addConversationFragment(**isInComingCall**);  
 } **else** {  
 **if** (getIntent() != **null** && getIntent().getExtras() != **null**) {  
 **isInComingCall** = getIntent().getExtras().getBoolean(Consts.***EXTRA\_IS\_INCOMING\_CALL***, **false**);  
 } **else** {  
 **isInComingCall** = **sharedPrefsHelper**.get(Consts.***EXTRA\_IS\_INCOMING\_CALL***, **false**);  
 }  
  
 **if** (!**isInComingCall**) {  
 **callService**.playRingtone();  
 }  
 startSuitableFragment(**isInComingCall**);  
 }  
 }  
  
 **private void** addListeners() {  
 addSessionEventsListener(**this**);  
 addSessionStateListener(**this**);  
  
 **connectionListener** = **new** ConnectionListenerImpl();  
 addConnectionListener(**connectionListener**);  
 }  
  
 **private void** removeListeners() {  
 removeSessionEventsListener(**this**);  
 removeSessionStateListener(**this**);  
 removeConnectionListener(**connectionListener**);  
 **callService**.removeCallTimerCallback();  
 }  
  
 **private void** bindCallService() {  
 **callServiceConnection** = **new** CallServiceConnection();  
 Intent intent = **new** Intent(**this**, CallService.**class**);  
 bindService(intent, **callServiceConnection**, Context.***BIND\_AUTO\_CREATE***);  
 }  
  
 @Override  
 **protected void** onActivityResult(**int** requestCode, **int** resultCode, **final** Intent data) {  
 Log.*i*(***TAG***, **"onActivityResult requestCode="** + requestCode + **", resultCode= "** + resultCode);  
 **if** (resultCode == Consts.***EXTRA\_LOGIN\_RESULT\_CODE***) {  
 **if** (data != **null**) {  
 **boolean** isLoginSuccess = data.getBooleanExtra(Consts.***EXTRA\_LOGIN\_RESULT***, **false**);  
 **if** (isLoginSuccess) {  
 initScreen();  
 } **else** {  
 CallService.*stop*(**this**);  
 finish();  
 }  
 }  
 }  
 **if** (requestCode == QBRTCScreenCapturer.***REQUEST\_MEDIA\_PROJECTION*** && resultCode == Activity.***RESULT\_OK*** && data != **null**) {  
 startScreenSharing(data);  
 Log.*i*(***TAG***, **"Starting Screen Capture"**);  
 }  
 }  
  
 **private void** startScreenSharing(**final** Intent data) {  
 Fragment fragmentByTag = getSupportFragmentManager().findFragmentByTag(ScreenShareFragment.**class**.getSimpleName());  
 **if** (!(fragmentByTag **instanceof** ScreenShareFragment)) {  
 ScreenShareFragment screenShareFragment = ScreenShareFragment.*newInstance*();  
 FragmentExecuotr.*addFragment*(getSupportFragmentManager(), R.id.***fragment\_container***, screenShareFragment, ScreenShareFragment.**class**.getSimpleName());  
  
 **if** (data != **null**) {  
 **callService**.startScreenSharing(data);  
 }  
 }  
 }  
  
 **private void** startSuitableFragment(**boolean** isInComingCall) {  
 QBRTCSession session = WebRtcSessionManager.*getInstance*(**this**).getCurrentSession();  
 **if** (session != **null**) {  
 **if** (isInComingCall) {  
 initIncomingCallTask();  
 startLoadAbsentUsers();  
 addIncomeCallFragment();  
 checkPermission();  
 } **else** {  
 addConversationFragment(isInComingCall);  
 getIntent().removeExtra(Consts.***EXTRA\_IS\_INCOMING\_CALL***);  
 **sharedPrefsHelper**.save(Consts.***EXTRA\_IS\_INCOMING\_CALL***, **false**);  
 }  
 } **else** {  
 finish();  
 }  
 }  
  
 **private void** checkPermission() {  
 **boolean** cam = SharedPrefsHelper.*getInstance*().get(Consts.***PERMISSIONS***[0], **true**);  
 **boolean** mic = SharedPrefsHelper.*getInstance*().get(Consts.***PERMISSIONS***[1], **true**);  
  
 **if** (**isVideoCall** && **checker**.lacksPermissions(Consts.***PERMISSIONS***)) {  
 **if** (cam) {  
 PermissionsActivity.*startActivity*(**this**, **false**, Consts.***PERMISSIONS***);  
 } **else** {  
 View rootView = getWindow().getDecorView().findViewById(android.R.id.***content***);  
 ErrorUtils.*showSnackbar*(rootView, R.string.***error\_permission\_video***, R.string.***dlg\_allow***, **new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 startPermissionSystemSettings();  
 }  
 });  
 }  
 } **else if** (**checker**.lacksPermissions(Consts.***PERMISSIONS***[1])) {  
 **if** (mic) {  
 PermissionsActivity.*startActivity*(**this**, **true**, Consts.***PERMISSIONS***);  
 } **else** {  
 View rootView = getWindow().getDecorView().findViewById(android.R.id.***content***);  
 ErrorUtils.*showSnackbar*(rootView, R.string.***error\_permission\_audio***, R.string.***dlg\_allow***, **new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 startPermissionSystemSettings();  
 }  
 });  
 }  
 }  
 }  
  
 **private void** startPermissionSystemSettings() {  
 Intent intent = **new** Intent(Settings.***ACTION\_APPLICATION\_DETAILS\_SETTINGS***);  
 Uri uri = Uri.*fromParts*(**"package"**, getPackageName(), **null**);  
 intent.setData(uri);  
 startActivityForResult(intent, ***REQUEST\_PERMISSION\_SETTING***);  
 }  
  
 **private void** startLoadAbsentUsers() {  
 ArrayList<QBUser> usersFromDb = **dbManager**.getAllUsers();  
 ArrayList<Integer> allParticipantsOfCall = **new** ArrayList<>();  
  
 **if** (**opponentsIdsList** != **null**) {  
 allParticipantsOfCall.addAll(**opponentsIdsList**);  
 }  
  
 **if** (**isInComingCall**) {  
 Integer callerID = **callService**.getCallerId();  
 **if** (callerID != **null**) {  
 allParticipantsOfCall.add(callerID);  
 }  
 }  
  
 ArrayList<Integer> idsUsersNeedLoad = UsersUtils.*getIdsNotLoadedUsers*(usersFromDb, allParticipantsOfCall);  
 **if** (!idsUsersNeedLoad.isEmpty()) {  
 **requestExecutor**.loadUsersByIds(idsUsersNeedLoad, **new** QBEntityCallbackImpl<ArrayList<QBUser>>() {  
 @Override  
 **public void** onSuccess(ArrayList<QBUser> users, Bundle params) {  
 **dbManager**.saveAllUsers(users, **false**);  
 notifyCallStateListenersNeedUpdateOpponentsList(users);  
 }  
 });  
 }  
 }  
  
 **private void** initSettingsStrategy() {  
 **if** (**opponentsIdsList** != **null**) {  
 SettingsUtil.*setSettingsStrategy*(**opponentsIdsList**, **sharedPref**, **this**);  
 }  
 }  
  
 **private void** initIncomingCallTask() {  
 **showIncomingCallWindowTaskHandler** = **new** Handler(Looper.*myLooper*());  
 **showIncomingCallWindowTask** = **new** Runnable() {  
 @Override  
 **public void** run() {  
 **if** (**callService**.currentSessionExist()) {  
 BaseSession.QBRTCSessionState currentSessionState = **callService**.getCurrentSessionState();  
 **if** (QBRTCSession.QBRTCSessionState.***QB\_RTC\_SESSION\_NEW***.equals(currentSessionState)) {  
 **callService**.rejectCurrentSession(**new** HashMap<>());  
 } **else** {  
 **callService**.stopRingtone();  
 hangUpCurrentSession();  
 }  
 }  
 ToastUtils.*longToast*(**"Call was stopped by timer"**);  
 }  
 };  
 }  
  
 **public void** hangUpCurrentSession() {  
 **callService**.stopRingtone();  
 **if** (!**callService**.hangUpCurrentSession(**new** HashMap<>())) {  
 CallService.*stop*(**this**);  
 finish();  
 }  
 }  
  
  
 **private void** startIncomeCallTimer(**long** time) {  
 Log.*d*(***TAG***, **"startIncomeCallTimer"**);  
 **showIncomingCallWindowTaskHandler**.postAtTime(**showIncomingCallWindowTask**, SystemClock.*uptimeMillis*() + time);  
 }  
  
 **private void** stopIncomeCallTimer() {  
 Log.*d*(***TAG***, **"stopIncomeCallTimer"**);  
 **showIncomingCallWindowTaskHandler**.removeCallbacks(**showIncomingCallWindowTask**);  
 }  
  
 @Override  
 **protected void** onResume() {  
 **super**.onResume();  
 bindCallService();  
 }  
  
 @Override  
 **protected void** onPause() {  
 **super**.onPause();  
 unbindService(**callServiceConnection**);  
 **if** (**callService** != **null**) {  
 removeListeners();  
 }  
 }  
  
 @Override  
 **public void** finish() {  
 *//Fix bug when user returns to call from service and the backstack doesn't have any screens* CallService.*stop*(**this**);  
 OpponentsActivity.*start*(**this**);  
 **super**.finish();  
 }  
  
 @Override  
 **public void** onBackPressed() {  
 *// To prevent returning from Call Fragment* }  
  
 **private void** addIncomeCallFragment() {  
 Log.*d*(***TAG***, **"Adding IncomeCallFragment"**);  
 **if** (**callService**.currentSessionExist()) {  
 IncomeCallFragment fragment = **new** IncomeCallFragment();  
 FragmentExecuotr.*addFragment*(getSupportFragmentManager(), R.id.***fragment\_container***, fragment, ***INCOME\_CALL\_FRAGMENT***);  
 } **else** {  
 Log.*d*(***TAG***, **"SKIP Adding IncomeCallFragment"**);  
 }  
 }  
  
 **private void** addConversationFragment(**boolean** isIncomingCall) {  
 BaseConversationFragment conversationFragment = BaseConversationFragment.*newInstance*(  
 **isVideoCall** ? **new** VideoConversationFragment()  
 : **new** AudioConversationFragment(),  
 isIncomingCall);  
 FragmentExecuotr.*addFragment*(getSupportFragmentManager(), R.id.***fragment\_container***, conversationFragment, conversationFragment.getClass().getSimpleName());  
 }  
  
 **private void** showNotificationPopUp(**final int** text, **final boolean** show) {  
 runOnUiThread(**new** Runnable() {  
 @Override  
 **public void** run() {  
 LinearLayout connectionView = (LinearLayout) View.*inflate*(CallActivity.**this**, R.layout.***connection\_popup***, **null**);  
 **if** (show) {  
 ((TextView) connectionView.findViewById(R.id.***notification***)).setText(text);  
 **if** (connectionView.getParent() == **null**) {  
 ((ViewGroup) CallActivity.**this**.findViewById(R.id.***fragment\_container***)).addView(connectionView);  
 }  
 } **else** {  
 ((ViewGroup) CallActivity.**this**.findViewById(R.id.***fragment\_container***)).removeView(connectionView);  
 }  
 }  
 });  
 }  
  
 *////////////////////////////// ConnectionListener //////////////////////////////* **private class** ConnectionListenerImpl **extends** AbstractConnectionListener {  
 @Override  
 **public void** connectionClosedOnError(Exception e) {  
 showNotificationPopUp(R.string.***connection\_was\_lost***, **true**);  
 }  
  
 @Override  
 **public void** reconnectionSuccessful() {  
 showNotificationPopUp(R.string.***connection\_was\_lost***, **false**);  
 }  
 }  
  
 *////////////////////////////// QBRTCSessionStateCallbackListener ///////////////////////////* @Override  
 **public void** onDisconnectedFromUser(QBRTCSession session, Integer userID) {  
 Log.*d*(***TAG***, **"Disconnected from user: "** + userID);  
 }  
  
 @Override  
 **public void** onConnectedToUser(QBRTCSession session, **final** Integer userID) {  
 notifyCallStateListenersCallStarted();  
 **if** (**isInComingCall**) {  
 stopIncomeCallTimer();  
 }  
 Log.*d*(***TAG***, **"onConnectedToUser() is started"**);  
 }  
  
 @Override  
 **public void** onConnectionClosedForUser(QBRTCSession session, Integer userID) {  
 Log.*d*(***TAG***, **"Connection closed for user: "** + userID);  
 }  
  
 @Override  
 **public void** onStateChanged(QBRTCSession qbrtcSession, BaseSession.QBRTCSessionState qbrtcSessionState) {  
  
 }  
  
 *////////////////////////////// QBRTCClientSessionCallbacks //////////////////////////////* @Override  
 **public void** onUserNotAnswer(QBRTCSession session, Integer userID) {  
 **if** (**callService**.isCurrentSession(session)) {  
 **callService**.stopRingtone();  
 }  
 }  
  
 @Override  
 **public void** onSessionStartClose(**final** QBRTCSession session) {  
 **if** (**callService**.isCurrentSession(session)) {  
 **callService**.removeSessionStateListener(**this**);  
 notifyCallStateListenersCallStopped();  
 }  
 }  
  
 @Override  
 **public void** onReceiveHangUpFromUser(**final** QBRTCSession session, **final** Integer userID, Map<String, String> map) {  
 **if** (**callService**.isCurrentSession(session)) {  
 **if** (userID.equals(session.getCallerID())) {  
 hangUpCurrentSession();  
 Log.*d*(***TAG***, **"Initiator hung up the call"**);  
 }  
 QBUser participant = **dbManager**.getUserById(userID);  
 **final** String participantName = participant != **null** ? participant.getFullName() : String.*valueOf*(userID);  
 ToastUtils.*shortToast*(**"User "** + participantName + **" "** + getString(R.string.***text\_status\_hang\_up***) + **" conversation"**);  
 }  
 }  
  
 @Override  
 **public void** onCallAcceptByUser(QBRTCSession session, Integer userId, Map<String, String> userInfo) {  
 **if** (**callService**.isCurrentSession(session)) {  
 **callService**.stopRingtone();  
 }  
 }  
  
 @Override  
 **public void** onReceiveNewSession(**final** QBRTCSession session) {  
 Log.*d*(***TAG***, **"Session "** + session.getSessionID() + **" Received"**);  
 }  
  
  
 @Override  
 **public void** onUserNoActions(QBRTCSession qbrtcSession, Integer integer) {  
 startIncomeCallTimer(0);  
 }  
  
 @Override  
 **public void** onSessionClosed(**final** QBRTCSession session) {  
 **if** (**callService**.isCurrentSession(session)) {  
 Log.*d*(***TAG***, **"Stopping session"**);  
 **callService**.stopForeground(**true**);  
 finish();  
 }  
 }  
  
 @Override  
 **public void** onCallRejectByUser(QBRTCSession session, Integer userID, Map<String, String> userInfo) {  
 **if** (**callService**.isCurrentSession(session)) {  
 **callService**.stopRingtone();  
 }  
 }  
  
 *////////////////////////////// IncomeCallFragmentCallbackListener ////////////////////////////* @Override  
 **public void** onAcceptCurrentSession() {  
 **if** (**callService**.currentSessionExist()) {  
 addConversationFragment(**true**);  
 } **else** {  
 Log.*d*(***TAG***, **"SKIP addConversationFragment method"**);  
 }  
 }  
  
 @Override  
 **public void** onRejectCurrentSession() {  
 **callService**.rejectCurrentSession(**new** HashMap<>());  
 }  
  
 *////////////////////////////// ConversationFragmentCallback ////////////////////////////* @Override  
 **public void** addConnectionListener(ConnectionListener connectionCallback) {  
 **callService**.addConnectionListener(connectionCallback);  
 }  
  
 @Override  
 **public void** removeConnectionListener(ConnectionListener connectionCallback) {  
 **callService**.removeConnectionListener(connectionCallback);  
 }  
  
 @Override  
 **public void** addSessionStateListener(QBRTCSessionStateCallback clientConnectionCallbacks) {  
 **callService**.addSessionStateListener(clientConnectionCallbacks);  
 }  
  
 @Override  
 **public void** addSessionEventsListener(QBRTCSessionEventsCallback eventsCallback) {  
 **callService**.addSessionEventsListener(eventsCallback);  
 }  
  
 @Override  
 **public void** onSetAudioEnabled(**boolean** isAudioEnabled) {  
 **callService**.setAudioEnabled(isAudioEnabled);  
 }  
  
 @Override  
 **public void** onHangUpCurrentSession() {  
 hangUpCurrentSession();  
 }  
  
 @TargetApi(21)  
 @Override  
 **public void** onStartScreenSharing() {  
 **if** (Build.VERSION.***SDK\_INT*** < Build.VERSION\_CODES.***LOLLIPOP***) {  
 **return**;  
 }  
 QBRTCScreenCapturer.*requestPermissions*(**this**);  
 }  
  
 @Override  
 **public void** onSwitchCamera(CameraVideoCapturer.CameraSwitchHandler cameraSwitchHandler) {  
 **callService**.switchCamera(cameraSwitchHandler);  
 }  
  
 @Override  
 **public void** onSetVideoEnabled(**boolean** isNeedEnableCam) {  
 **callService**.setVideoEnabled(isNeedEnableCam);  
 }  
  
 @Override  
 **public void** onSwitchAudio() {  
 **callService**.switchAudio();  
 }  
  
 @Override  
 **public void** removeSessionStateListener(QBRTCSessionStateCallback clientConnectionCallbacks) {  
 **callService**.removeSessionStateListener(clientConnectionCallbacks);  
 }  
  
 @Override  
 **public void** removeSessionEventsListener(QBRTCSessionEventsCallback eventsCallback) {  
 **callService**.removeSessionEventsListener(eventsCallback);  
 }  
  
 @Override  
 **public void** addCurrentCallStateListener(CurrentCallStateCallback currentCallStateCallback) {  
 **if** (currentCallStateCallback != **null**) {  
 **currentCallStateCallbackList**.add(currentCallStateCallback);  
 }  
 }  
  
 @Override  
 **public void** removeCurrentCallStateListener(CurrentCallStateCallback currentCallStateCallback) {  
 **currentCallStateCallbackList**.remove(currentCallStateCallback);  
 }  
  
 @Override  
 **public void** addOnChangeAudioDeviceListener(OnChangeAudioDevice onChangeDynamicCallback) {  
  
 }  
  
 @Override  
 **public void** removeOnChangeAudioDeviceListener(OnChangeAudioDevice onChangeDynamicCallback) {  
  
 }  
  
 @Override  
 **public void** acceptCall(Map<String, String> userInfo) {  
 **callService**.acceptCall(userInfo);  
 }  
  
 @Override  
 **public void** startCall(Map<String, String> userInfo) {  
 **callService**.startCall(userInfo);  
 }  
  
 @Override  
 **public boolean** currentSessionExist() {  
 **return callService**.currentSessionExist();  
 }  
  
 @Override  
 **public** List<Integer> getOpponents() {  
 **return callService**.getOpponents();  
 }  
  
 @Override  
 **public** Integer getCallerId() {  
 **return callService**.getCallerId();  
 }  
  
 @Override  
 **public void** addVideoTrackListener(QBRTCClientVideoTracksCallbacks<QBRTCSession> callback) {  
 **callService**.addVideoTrackListener(callback);  
 }  
  
 @Override  
 **public void** removeVideoTrackListener(QBRTCClientVideoTracksCallbacks<QBRTCSession> callback) {  
 **callService**.removeVideoTrackListener(callback);  
 }  
  
 @Override  
 **public** BaseSession.QBRTCSessionState getCurrentSessionState() {  
 **return callService**.getCurrentSessionState();  
 }  
  
 @Override  
 **public** QBRTCTypes.QBRTCConnectionState getPeerChannel(Integer userId) {  
 **return callService**.getPeerChannel(userId);  
 }  
  
 @Override  
 **public boolean** isMediaStreamManagerExist() {  
 **return callService**.isMediaStreamManagerExist();  
 }  
  
 @Override  
 **public boolean** isCallState() {  
 **return callService**.isCallMode();  
 }  
  
 @Override  
 **public** HashMap<Integer, QBRTCVideoTrack> getVideoTrackMap() {  
 **return callService**.getVideoTrackMap();  
 }  
  
 @Override  
 **public** QBRTCVideoTrack getVideoTrack(Integer userId) {  
 **return callService**.getVideoTrack(userId);  
 }  
  
 @Override  
 **public void** onStopPreview() {  
 **callService**.stopScreenSharing();  
 addConversationFragment(**false**);  
 }  
  
  
 **private void** notifyCallStateListenersCallStarted() {  
 **for** (CurrentCallStateCallback callback : **currentCallStateCallbackList**) {  
 callback.onCallStarted();  
 }  
 }  
  
 **private void** notifyCallStateListenersCallStopped() {  
 **for** (CurrentCallStateCallback callback : **currentCallStateCallbackList**) {  
 callback.onCallStopped();  
 }  
 }  
  
 **private void** notifyCallStateListenersNeedUpdateOpponentsList(**final** ArrayList<QBUser> newUsers) {  
 **for** (CurrentCallStateCallback callback : **currentCallStateCallbackList**) {  
 callback.onOpponentsListUpdated(newUsers);  
 }  
 }  
  
 **private void** notifyCallStateListenersCallTime(String callTime) {  
 **for** (CurrentCallStateCallback callback : **currentCallStateCallbackList**) {  
 callback.onCallTimeUpdate(callTime);  
 }  
 }  
  
 **private class** CallServiceConnection **implements** ServiceConnection {  
 @Override  
 **public void** onServiceDisconnected(ComponentName name) {  
  
 }  
  
 @Override  
 **public void** onServiceConnected(ComponentName name, IBinder service) {  
 CallService.CallServiceBinder binder = (CallService.CallServiceBinder) service;  
 **callService** = binder.getService();  
 **if** (**callService**.currentSessionExist()) {  
 *//we have already currentSession == null, so it's no reason to do further initialization* **if** (QBChatService.*getInstance*().isLoggedIn()) {  
 initScreen();  
 } **else** {  
 login();  
 }  
 } **else** {  
 finish();  
 }  
 }  
  
 **private void** login() {  
 QBUser qbUser = SharedPrefsHelper.*getInstance*().getQbUser();  
 Intent tempIntent = **new** Intent(CallActivity.**this**, LoginService.**class**);  
 PendingIntent pendingIntent = createPendingResult(Consts.***EXTRA\_LOGIN\_RESULT\_CODE***, tempIntent, 0);  
 LoginService.*start*(CallActivity.**this**, qbUser, pendingIntent);  
 }  
 }  
  
 **private class** CallTimerCallback **implements** CallService.CallTimerListener {  
 @Override  
 **public void** onCallTimeUpdate(String time) {  
 runOnUiThread(**new** Runnable() {  
 @Override  
 **public void** run() {  
 notifyCallStateListenersCallTime(time);  
 }  
 });  
 }  
 }  
  
 **public interface** OnChangeAudioDevice {  
 **void** audioDeviceChanged(AppRTCAudioManager.AudioDevice newAudioDevice);  
 }  
  
  
 **public interface** CurrentCallStateCallback {  
 **void** onCallStarted();  
  
 **void** onCallStopped();  
  
 **void** onOpponentsListUpdated(ArrayList<QBUser> newUsers);  
  
 **void** onCallTimeUpdate(String time);  
 }  
}

opponentacvtivity

**package** com.quickblox.sample.videochat.java.activities;  
  
**import** android.app.ActivityManager;  
**import** android.app.NotificationManager;  
**import** android.content.Context;  
**import** android.content.Intent;  
**import** android.os.Bundle;  
**import** android.util.Log;  
**import** android.view.Menu;  
**import** android.view.MenuItem;  
  
**import** com.quickblox.chat.QBChatService;  
**import** com.quickblox.core.QBEntityCallback;  
**import** com.quickblox.core.exception.QBResponseException;  
**import** com.quickblox.core.request.GenericQueryRule;  
**import** com.quickblox.core.request.QBPagedRequestBuilder;  
**import** com.quickblox.messages.services.SubscribeService;  
**import** com.quickblox.sample.videochat.java.R;  
**import** com.quickblox.sample.videochat.java.adapters.UsersAdapter;  
**import** com.quickblox.sample.videochat.java.db.QbUsersDbManager;  
**import** com.quickblox.sample.videochat.java.services.CallService;  
**import** com.quickblox.sample.videochat.java.services.LoginService;  
**import** com.quickblox.sample.videochat.java.utils.CollectionsUtils;  
**import** com.quickblox.sample.videochat.java.utils.Consts;  
**import** com.quickblox.sample.videochat.java.utils.PermissionsChecker;  
**import** com.quickblox.sample.videochat.java.utils.PushNotificationSender;  
**import** com.quickblox.sample.videochat.java.utils.SharedPrefsHelper;  
**import** com.quickblox.sample.videochat.java.utils.ToastUtils;  
**import** com.quickblox.sample.videochat.java.utils.UsersUtils;  
**import** com.quickblox.sample.videochat.java.utils.WebRtcSessionManager;  
**import** com.quickblox.users.model.QBUser;  
**import** com.quickblox.videochat.webrtc.QBRTCClient;  
**import** com.quickblox.videochat.webrtc.QBRTCSession;  
**import** com.quickblox.videochat.webrtc.QBRTCTypes;  
  
**import** java.util.ArrayList;  
**import** java.util.List;  
  
**import** androidx.recyclerview.widget.LinearLayoutManager;  
**import** androidx.recyclerview.widget.RecyclerView;  
  
*/\*\*  
 \* QuickBlox team  
 \*/***public class** OpponentsActivity **extends** BaseActivity {  
 **private static final** String ***TAG*** = OpponentsActivity.**class**.getSimpleName();  
  
 **private static final int *PER\_PAGE\_SIZE\_100*** = 100;  
 **private static final** String ***ORDER\_RULE*** = **"order"**;  
 **private static final** String ***ORDER\_DESC\_UPDATED*** = **"desc date updated\_at"**;  
  
 **private** RecyclerView **usersRecyclerview**;  
 **private** QBUser **currentUser**;  
 **private** UsersAdapter **usersAdapter**;  
  
 **private** QbUsersDbManager **dbManager**;  
 **private** PermissionsChecker **checker**;  
  
 **public static void** start(Context context) {  
 Intent intent = **new** Intent(context, OpponentsActivity.**class**);  
 intent.addFlags(Intent.***FLAG\_ACTIVITY\_CLEAR\_TASK***);  
 context.startActivity(intent);  
 }  
  
 @Override  
 **public void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_select\_users***);  
 **currentUser** = SharedPrefsHelper.*getInstance*().getQbUser();  
 **dbManager** = QbUsersDbManager.*getInstance*(getApplicationContext());  
 **checker** = **new** PermissionsChecker(getApplicationContext());  
  
 initDefaultActionBar();  
 initUi();  
 startLoginService();  
 }  
  
 @Override  
 **protected void** onResume() {  
 **super**.onResume();  
 **boolean** isIncomingCall = SharedPrefsHelper.*getInstance*().get(Consts.***EXTRA\_IS\_INCOMING\_CALL***, **false**);  
 **if** (isCallServiceRunning(CallService.**class**)) {  
 Log.*d*(***TAG***, **"CallService is running now"**);  
 CallActivity.*start*(**this**, isIncomingCall);  
 }  
 clearAppNotifications();  
 loadUsers();  
 }  
  
 **private boolean** isCallServiceRunning(Class<?> serviceClass) {  
 ActivityManager manager = (ActivityManager) getSystemService(Context.***ACTIVITY\_SERVICE***);  
 **for** (ActivityManager.RunningServiceInfo service : manager.getRunningServices(Integer.***MAX\_VALUE***)) {  
 **if** (serviceClass.getName().equals(service.**service**.getClassName())) {  
 **return true**;  
 }  
 }  
 **return false**;  
 }  
  
 **private void** clearAppNotifications() {  
 NotificationManager notificationManager = (NotificationManager) getSystemService(Context.***NOTIFICATION\_SERVICE***);  
 **if** (notificationManager != **null**) {  
 notificationManager.cancelAll();  
 }  
 }  
  
 **private void** startPermissionsActivity(**boolean** checkOnlyAudio) {  
 PermissionsActivity.*startActivity*(**this**, checkOnlyAudio, Consts.***PERMISSIONS***);  
 }  
  
 **private void** loadUsers() {  
 showProgressDialog(R.string.***dlg\_loading\_opponents***);  
 ArrayList<GenericQueryRule> rules = **new** ArrayList<>();  
 rules.add(**new** GenericQueryRule(***ORDER\_RULE***, ***ORDER\_DESC\_UPDATED***));  
  
 QBPagedRequestBuilder qbPagedRequestBuilder = **new** QBPagedRequestBuilder();  
 qbPagedRequestBuilder.setRules(rules);  
 qbPagedRequestBuilder.setPerPage(***PER\_PAGE\_SIZE\_100***);  
  
 **requestExecutor**.loadLastUpdatedUsers(qbPagedRequestBuilder, **new** QBEntityCallback<ArrayList<QBUser>>() {  
 @Override  
 **public void** onSuccess(ArrayList<QBUser> qbUsers, Bundle bundle) {  
 Log.*d*(***TAG***, **"Successfully loaded Last 100 created users"**);  
 **dbManager**.saveAllUsers(qbUsers, **true**);  
 initUsersList();  
 hideProgressDialog();  
 }  
  
 @Override  
 **public void** onError(QBResponseException e) {  
 Log.*d*(***TAG***, **"Error load users"** + e.getMessage());  
 hideProgressDialog();  
 showErrorSnackbar(R.string.***loading\_users\_error***, e, v -> loadUsers());  
 }  
 });  
 }  
  
 **private void** initUi() {  
 **usersRecyclerview** = findViewById(R.id.***list\_select\_users***);  
 }  
  
 **private void** initUsersList() {  
 List<QBUser> currentOpponentsList = **dbManager**.getAllUsers();  
 Log.*d*(***TAG***, **"initUsersList currentOpponentsList= "** + currentOpponentsList);  
 currentOpponentsList.remove(**sharedPrefsHelper**.getQbUser());  
 **if** (**usersAdapter** == **null**) {  
 **usersAdapter** = **new** UsersAdapter(**this**, currentOpponentsList);  
 **usersAdapter**.setSelectedItemsCountsChangedListener(**new** UsersAdapter.SelectedItemsCountChangedListener() {  
 @Override  
 **public void** onCountSelectedItemsChanged(Integer count) {  
 updateActionBar(count);  
 }  
 });  
  
 **usersRecyclerview**.setLayoutManager(**new** LinearLayoutManager(**this**));  
 **usersRecyclerview**.setAdapter(**usersAdapter**);  
 } **else** {  
 **usersAdapter**.updateUsersList(currentOpponentsList);  
 }  
 }  
  
 @Override  
 **public boolean** onCreateOptionsMenu(Menu menu) {  
 **if** (**usersAdapter** != **null** && !**usersAdapter**.getSelectedUsers().isEmpty()) {  
 getMenuInflater().inflate(R.menu.***activity\_selected\_opponents***, menu);  
 } **else** {  
 getMenuInflater().inflate(R.menu.***activity\_opponents***, menu);  
 }  
  
 **return super**.onCreateOptionsMenu(menu);  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **int** id = item.getItemId();  
  
 **switch** (id) {  
 **case** R.id.***update\_opponents\_list***:  
 loadUsers();  
 **return true**;  
  
 **case** R.id.***settings***:  
 SettingsActivity.*start*(**this**);  
 **return true**;  
  
 **case** R.id.***log\_out***:  
 logOut();  
 **return true**;  
  
 **case** R.id.***start\_video\_call***:  
 **if** (checkIsLoggedInChat()) {  
 startCall(**true**);  
 }  
 **if** (**checker**.lacksPermissions(Consts.***PERMISSIONS***)) {  
 startPermissionsActivity(**false**);  
 }  
 **return true**;  
  
 **case** R.id.***start\_audio\_call***:  
 **if** (checkIsLoggedInChat()) {  
 startCall(**false**);  
 }  
 **if** (**checker**.lacksPermissions(Consts.***PERMISSIONS***[1])) {  
 startPermissionsActivity(**true**);  
 }  
 **return true**;  
 **case** R.id.***menu\_appinfo***:  
 AppInfoActivity.*start*(**this**);  
 **return true**;  
  
 **default**:  
 **return super**.onOptionsItemSelected(item);  
 }  
 }  
  
 **private boolean** checkIsLoggedInChat() {  
 **if** (!QBChatService.*getInstance*().isLoggedIn()) {  
 startLoginService();  
 ToastUtils.*shortToast*(R.string.***dlg\_relogin\_wait***);  
 **return false**;  
 }  
 **return true**;  
 }  
  
 **private void** startLoginService() {  
 **if** (**sharedPrefsHelper**.hasQbUser()) {  
 QBUser qbUser = **sharedPrefsHelper**.getQbUser();  
 LoginService.*start*(**this**, qbUser);  
 }  
 }  
  
 **private void** startCall(**boolean** isVideoCall) {  
 Log.*d*(***TAG***, **"Starting Call"**);  
 **if** (**usersAdapter**.getSelectedUsers().size() > Consts.***MAX\_OPPONENTS\_COUNT***) {  
 ToastUtils.*longToast*(String.*format*(getString(R.string.***error\_max\_opponents\_count***),  
 Consts.***MAX\_OPPONENTS\_COUNT***));  
 **return**;  
 }  
  
 ArrayList<Integer> opponentsList = CollectionsUtils.*getIdsSelectedOpponents*(**usersAdapter**.getSelectedUsers());  
 QBRTCTypes.QBConferenceType conferenceType = isVideoCall  
 ? QBRTCTypes.QBConferenceType.***QB\_CONFERENCE\_TYPE\_VIDEO*** : QBRTCTypes.QBConferenceType.***QB\_CONFERENCE\_TYPE\_AUDIO***;  
 Log.*d*(***TAG***, **"conferenceType = "** + conferenceType);  
  
 QBRTCClient qbrtcClient = QBRTCClient.*getInstance*(getApplicationContext());  
 QBRTCSession newQbRtcSession = qbrtcClient.createNewSessionWithOpponents(opponentsList, conferenceType);  
 WebRtcSessionManager.*getInstance*(**this**).setCurrentSession(newQbRtcSession);  
 PushNotificationSender.*sendPushMessage*(opponentsList, **currentUser**.getFullName());  
 CallActivity.*start*(**this**, **false**);  
 }  
  
 **private void** updateActionBar(**int** countSelectedUsers) {  
 **if** (countSelectedUsers < 1) {  
 initDefaultActionBar();  
 } **else** {  
 removeActionbarSubTitle();  
 setActionBarTitle(String.format(getString(  
 countSelectedUsers > 1  
 ? R.string.tile\_many\_users\_selected  
 : R.string.title\_one\_user\_selected),  
 countSelectedUsers));  
 }  
  
 invalidateOptionsMenu();  
 }  
  
 **private void** logOut() {  
 Log.d(TAG, **"Removing User data, and Logout"**);  
 SubscribeService.unSubscribeFromPushes(**this**);  
 LoginService.logout(**this**);  
 UsersUtils.removeUserData(getApplicationContext());  
 requestExecutor.signOut();  
 startLoginActivity();  
 }  
  
 **private void** startLoginActivity() {  
 LoginActivity.start(**this**);  
 finish();  
 }  
}

JAVA : MODELS

MESSAGE

**package** com.mys3soft.mys3chat.Models;  
  
  
**public class** Message {  
  
 **public** String **FromMail**;  
 **public** String **ToMail**;  
 **public** String **Message**;  
 **public** String **SentDate**;  
 **public** String **FriendFullName**;  
  
 **public int rowid**;  
}

NotificationModel

**package** com.mys3soft.mys3chat.Models;  
  
  
*/\*  
NotificationType:  
 1) Friend Request  
  
  
  
 \*/***public class** NotificationModel {  
  
  
 **public** String **NotificationMessage**;  
 **public int NotificationType**;  
 **public** String **EmailFrom**;  
  
 **public** String **FirstName**;  
 **public** String **LastName**;  
  
 **public** String **FriendRequestFireBaseKey**;  
  
}

StaticInfo

**package** com.mys3soft.mys3chat.Models;  
  
  
**public class** StaticInfo {  
  
 **public static** String *EndPoint* = **"https://mys3chat.firebaseio.com"**;  
 **public static** String *MessagesEndPoint* = **"https://mys3chat.firebaseio.com/messages"**;  
 **public static** String *FriendsURL* = **"https://mys3chat.firebaseio.com/friends"**;  
 **public static** String *UsersURL* = **"https://mys3chat.firebaseio.com/users"**;  
 **public static** String *UserCurrentChatFriendEmail* = **""**;  
 **public static** String *TypingStatus* = **"TypingStatus"**;  
  
 **public static** String *NotificationEndPoint* = **"https://mys3chat.firebaseio.com/notifications"**;  
 **public static** String *FriendRequestsEndPoint* = **"https://mys3chat.firebaseio.com/friendrequests"**;  
  
 **public static int** *ChatAciviityRequestCode* = 101;  
  
}

User

**package** com.mys3soft.mys3chat.Models;  
  
  
**public class** User {  
  
 **public** String **Password**;  
  
  
 **public long ID**;  
 **public** String **Email**;  
 **public** String **FirstName**;  
 **public** String **LastName**;  
 **public** String **Entry date**;

**public** String **Mobilenumber**;

}

JAVA: Services

Data content

**package** com.mys3soft.mys3chat.Services;  
  
  
**import** android.content.Context;  
**import** android.database.Cursor;  
**import** android.database.SQLException;  
**import** android.database.sqlite.SQLiteDatabase;  
**import** android.database.sqlite.SQLiteOpenHelper;  
  
**import** com.mys3soft.mys3chat.Models.Message;  
**import** com.mys3soft.mys3chat.Models.User;  
  
**import** java.util.ArrayList;  
**import** java.util.Collections;  
**import** java.util.Comparator;  
**import** java.util.EnumMap;  
**import** java.util.List;  
  
  
*/\*  
Tables:  
 2) Friends -> contains local user friend list  
 3) Messages  
  
 \*/***public class** DataContext **extends** SQLiteOpenHelper {  
  
 **public** DataContext(Context context, String name, SQLiteDatabase.CursorFactory factory, **int** version) {  
 **super**(context, **"mys3chat.db"**, factory, 3);  
 }  
  
 @Override  
 **public void** onCreate(SQLiteDatabase db) {  
 *//String tblLocalUser = "create table if not exists LocalUser (ID integer ,Email text, FirstName text, LastName text); ";* String tblFriends = **"create table if not exists Friends (Email text, FirstName text, LastName text);"**;  
 String tblMessages = **"create table if not exists Messages (FromMail text, ToMail text, Message text, SentDate text);"**;  
 *//db.execSQL(tblLocalUser);* db.execSQL(tblFriends);  
 db.execSQL(tblMessages);  
 }  
  
 @Override  
 **public void** onUpgrade(SQLiteDatabase db, **int** oldVersion, **int** newVersion) {  
 *//String dropLocalUser = "drop table if exists LocalUser;";* String dropFriends = **"drop table if exists Friends; "**;  
 String dropMessages = **"drop table if exists Messages;"**;  
 *// db.execSQL(dropLocalUser);* db.execSQL(dropFriends);  
 db.execSQL(dropMessages);  
 onCreate(db);  
 }  
  
 **public** List<User> getUserFriendList() {  
 List<User> friendList = **new** ArrayList<>();  
 SQLiteDatabase db = getReadableDatabase();  
 String query = **"select \* from Friends"**;  
 Cursor c = db.rawQuery(query, **null**);  
 c.moveToFirst();  
 **while** (!c.isAfterLast()) {  
 **try** {  
 User friend = **new** User();  
 friend.**Email** = c.getString(c.getColumnIndex(**"Email"**));  
 friend.**FirstName** = c.getString(c.getColumnIndex(**"FirstName"**));  
 friend.**LastName** = c.getString(c.getColumnIndex(**"LastName"**));  
 friendList.add(friend);  
 c.moveToNext();  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 c.close();  
  
 Collections.*sort*(friendList, **new** Comparator<User>() {  
 @Override  
 **public int** compare(User o1, User o2) {  
 **return** o1.**FirstName**.compareTo(o2.**FirstName**);  
 }  
 });  
  
 **return** friendList;  
  
 }  
  
 **public void** refreshUserFriendList(List<User> friendList) {  
  
 **for** (User item : friendList) {  
 *// check if user already exists* **if** (checkFriendAlreadyExists(item.**Email**) == 0) {  
 *// insert* SQLiteDatabase db = getWritableDatabase();  
 String query = **"insert into Friends (Email,FirstName,LastName) values('"** + item.**Email** + **"', '"** + item.**FirstName** + **"', '"** + item.**LastName** + **"');"**;  
 db.execSQL(query);  
 *// db.close();* }  
 }  
 }  
  
 **public int** checkFriendAlreadyExists(String email) {  
 Cursor c = **null**;  
 SQLiteDatabase db = **null**;  
 **try** {  
 db = getReadableDatabase();  
 String query = **"select count(\*) from Friends where Email = '"** + email + **"'"**;  
 c = db.rawQuery(query, **null**);  
 **if** (c.moveToFirst()) {  
 **return** c.getInt(0);  
 }  
 **return** 0;  
 } **finally** {  
 **if** (c != **null**) {  
 c.close();  
 }  
 **if** (db != **null**) {  
 db.close();  
 }  
 }  
 }  
  
 **public void** deleteAllFriendsFromLocalDB() {  
 String query = **"delete from Friends"**;  
 *// String queryMess = "delete from Messages";* SQLiteDatabase db = getWritableDatabase();  
 db.execSQL(query);  
 *//db.execSQL(queryMess);* }  
  
 **public void** deleteFriendByEmailFromLocalDB(String email) {  
 String query = **"delete from Friends where Email = '"** + email + **"';"**;  
 getWritableDatabase().execSQL(query);  
 }  
  
 **public** User getFriendByEmailFromLocalDB(String friendEmail) {  
 String query = **"select \* from Friends where Email = '"** + friendEmail + **"';"**;  
 SQLiteDatabase db = getReadableDatabase();  
 Cursor c = db.rawQuery(query, **null**);  
 c.moveToFirst();  
 User friend = **new** User();  
  
 **if** (c.getCount() > 0) {  
 friend.**Email** = c.getString(c.getColumnIndex(**"Email"**));  
 friend.**FirstName** = c.getString(c.getColumnIndex(**"FirstName"**));  
 friend.**LastName** = c.getString(c.getColumnIndex(**"LastName"**));  
 }  
 **return** friend;  
 }  
  
 **public void** saveMessageOnLocakDB(String from, String to, String message, String sentDate) {  
 SQLiteDatabase db = getWritableDatabase();  
 String query = **"insert into Messages (FromMail, ToMail, Message, SentDate) values('"** + from + **"', '"** + to + **"', '"** + message.replace(**"'"**, **"\""**) + **"','"** + sentDate + **"');"**;  
 db.execSQL(query);  
 }  
  
 **public** List<Message> getChat(String userMail, String friendMail, **int** pageNo) {  
 List<Message> messageList = **new** ArrayList<>();  
 SQLiteDatabase db = getReadableDatabase();  
 **try** {  
 **int** limit = (5 \* pageNo) + 35;  
 String whereCondition = **"((FromMail = '"** + userMail + **"' and ToMail='"** + friendMail + **"') or (ToMail = '"** + userMail + **"' and FromMail='"** + friendMail + **"'))"**;  
 String query = **"select \* from ( select rowid, \* from Messages where "** + whereCondition + **" order by rowid desc limit "** + limit + **") order by rowid "**;  
 Cursor c = db.rawQuery(query, **null**);  
 c.moveToFirst();  
 **while** (!c.isAfterLast()) {  
 Message mess = **new** Message();  
 mess.**FromMail** = c.getString(c.getColumnIndex(**"FromMail"**));  
 mess.**ToMail** = c.getString(c.getColumnIndex(**"ToMail"**));  
 mess.**Message** = c.getString(c.getColumnIndex(**"Message"**));  
  
 mess.**SentDate** = c.getString(c.getColumnIndex(**"SentDate"**));  
 messageList.add(mess);  
 c.moveToNext();  
 }  
 c.close();  
 **return** messageList;  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 **return** messageList;  
 }  
  
 }  
  
 **public void** deleteChat(String userMail, String friendMail) {  
 String deleteQuery = **"delete from Messages where (FromMail = '"** + userMail + **"' and ToMail='"** + friendMail + **"') or (ToMail = '"** + userMail + **"' and FromMail='"** + friendMail + **"') "**;  
 getWritableDatabase().execSQL(deleteQuery);  
  
 }  
  
 **public** List<Message> getUserLastChatList(String userMail) {  
 List<User> userFriendList = getUserFriendList();  
 List<Message> userLastChat = **new** ArrayList<>();  
 SQLiteDatabase db = getReadableDatabase();  
 **for** (User friend : userFriendList) {  
 String query = **"select rowid, \* from Messages where (FromMail = '"** + userMail + **"' and ToMail='"** + friend.**Email** + **"') or (ToMail = '"** + userMail + **"' and FromMail='"** + friend.**Email** + **"') order by rowid desc limit 1 "**;  
 Cursor c = db.rawQuery(query, **null**);  
 c.moveToFirst();  
 **try** {  
 Message mess = **new** Message();  
 *// set from email to friend so when user click on list to navigate to chat activity* mess.**FromMail** = friend.**Email**;  
 mess.**Message** = c.getString(c.getColumnIndex(**"Message"**));  
 mess.**Message** = mess.**Message**.replace(**"\n"**, **""**);  
 mess.**SentDate** = c.getString(c.getColumnIndex(**"SentDate"**));  
 mess.**FriendFullName** = friend.**FirstName** + **" "** + friend.**LastName**;  
 mess.**rowid** = c.getInt(c.getColumnIndex(**"rowid"**));  
 userLastChat.add(mess);  
 } **catch** (Exception e) {  
  
 }  
  
 }  
 Collections.*sort*(userLastChat, **new** Comparator<Message>() {  
 @Override  
 **public int** compare(Message o1, Message o2) {  
 *// -1) Less Than 0) equal 1) Greater than* **return** o1.**rowid** > o2.**rowid** ? -1 : 1;  
 }  
 });  
  
 **return** userLastChat;  
 }  
  
 **public void** setPreferedDisplayName(String friendEmail, String newName) {  
 String query = **"update Friends set FirstName = '"** + newName + **"', LastName='' where Email='"** + friendEmail + **"' "**;  
 getWritableDatabase().execSQL(query);  
 }  
  
}

LocaluserService

**package** com.mys3soft.mys3chat.Services;  
  
**import** android.content.Context;  
**import** android.content.SharedPreferences;  
  
**import** com.mys3soft.mys3chat.Models.User;  
  
  
**public class** LocalUserService {  
 **public static** User getLocalUserFromPreferences(Context context){  
 SharedPreferences pref = context.getSharedPreferences(**"LocalUser"**,0);  
 User user = **new** User();  
 user.**Email** = pref.getString(**"Email"**,**null**);  
 user.**FirstName** = pref.getString(**"FirstName"**,**null**);  
 user.**LastName** = pref.getString(**"LastName"**,**null**);  
 **return** user;  
 }  
  
 **public static boolean** deleteLocalUserFromPreferences(Context context){  
 **try** {  
 SharedPreferences pref = context.getSharedPreferences(**"LocalUser"**,0);  
 SharedPreferences.Editor editor = pref.edit();  
 editor.clear();  
 editor.commit();  
 **return true**;  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 **return true**;  
 }  
 }  
  
  
  
}

Tools

**package** com.mys3soft.mys3chat.Services;  
  
**import** android.content.Context;  
**import** android.net.ConnectivityManager;  
**import** android.net.NetworkInfo;  
  
**import** java.text.SimpleDateFormat;  
**import** java.util.Calendar;  
**import** java.util.Date;  
**import** java.util.HashMap;  
**import** java.util.Map;  
  
**import** retrofit2.Retrofit;  
**import** retrofit2.converter.scalars.ScalarsConverterFactory;  
  
**public class** Tools {  
  
 **public static final** String ***ENDPOINT*** = **"https://mys3chat.firebaseio.com"**;  
  
 **public static** String encodeString(String string) {  
 **return** string.replace(**"."**, **","**);  
 }  
  
 **public static** String decodeString(String string) {  
 **return** string.replace(**","**, **"."**);  
 }  
  
 **public static** IFireBaseAPI makeRetroFitApi() {  
 Retrofit retrofit = **new** Retrofit.Builder()  
 .baseUrl(***ENDPOINT***)  
 .addConverterFactory(ScalarsConverterFactory.*create*())  
 .build();  
 **return** retrofit.create(IFireBaseAPI.**class**);  
 }  
  
 **public static** String toProperName(String s) {  
 **if** (s.length() <= 1)  
 **return** s.toUpperCase();  
 **return** s.substring(0, 1).toUpperCase() + s.substring(1).toLowerCase();  
 }  
  
 **public static int** createUniqueIdPerUser(String userEmail) {  
 String email = userEmail.split(**"@"**)[0].toLowerCase().replaceAll(**"[^a-zA-Z0-9]"**, **""**);  
 **final** Map<Character, Integer> map;  
 map = **new** HashMap<>();  
 map.put(**'a'**, 1);  
 map.put(**'b'**, 2);  
 map.put(**'c'**, 3);  
 map.put(**'d'**, 4);  
 map.put(**'e'**, 5);  
 map.put(**'f'**, 6);  
 map.put(**'g'**, 7);  
 map.put(**'h'**, 8);  
 map.put(**'i'**, 9);  
 map.put(**'j'**, 10);  
 map.put(**'k'**, 11);  
 map.put(**'l'**, 12);  
 map.put(**'m'**, 13);  
 map.put(**'n'**, 14);  
 map.put(**'o'**, 15);  
 map.put(**'p'**, 16);  
 map.put(**'q'**, 17);  
 map.put(**'r'**, 18);  
 map.put(**'s'**, 19);  
 map.put(**'t'**, 20);  
 map.put(**'u'**, 21);  
 map.put(**'v'**, 22);  
 map.put(**'w'**, 23);  
 map.put(**'x'**, 24);  
 map.put(**'y'**, 25);  
 map.put(**'z'**, 26);  
 String intEmail = **""**;  
  
 **for** (**char** c : email.toCharArray()) {  
 **int** val = 0;  
 **try** {  
 val = map.get(c);  
 } **catch** (Exception e) {  
  
 }  
 intEmail += val;  
 }  
  
 **if** (intEmail.length() > 9) {  
 intEmail = intEmail.substring(0, 9);  
 }  
  
 **return** Integer.*parseInt*(intEmail);  
  
 }  
  
 **public static boolean** isValidEmail(String email) {  
 String emailPattern = **"[a-zA-Z0-9.\_-]+@[a-z]+\\.+[a-zA-Z0-9.\_-]+"**;  
 **return** email.matches(emailPattern);  
 }  
  
 **public static** String toCharacterMonth(**int** month) {  
 **if** (month == 1) **return "Jan"**;  
 **else if** (month == 2) **return "Feb"**;  
 **else if** (month == 3) **return "Mar"**;  
 **else if** (month == 4) **return "Apr"**;  
 **else if** (month == 5) **return "May"**;  
 **else if** (month == 6) **return "Jun"**;  
 **else if** (month == 7) **return "Jul"**;  
 **else if** (month == 8) **return "Aug"**;  
 **else if** (month == 9) **return "Sep"**;  
 **else if** (month == 10) **return "Oct"**;  
 **else if** (month == 11) **return "Nov"**;  
 **else return "Dec"**;  
 }  
  
 **public static** String lastSeenProper(String lastSeenDate) {  
 SimpleDateFormat dateFormat = **new** SimpleDateFormat(**"dd MM yy hh:mm a"**);  
 Date currentDate = **new** Date();  
 String cuurentDateString = dateFormat.format(currentDate);  
 Date nw = **null**;  
 Date seen = **null**;  
 **try** {  
 nw = dateFormat.parse(cuurentDateString);  
 seen = dateFormat.parse(lastSeenDate);  
 **long** diff = nw.getTime() - seen.getTime();  
 **long** diffDays = diff / (24 \* 60 \* 60 \* 1000);  
 **long** diffHours = diff / (60 \* 60 \* 1000) % 24;  
 **long** diffMinutes = diff / (60 \* 1000) % 60;  
 **if** (diffDays > 0) {  
 String[] originalDate = lastSeenDate.split(**" "**);  
 **return "Last seen "** + originalDate[0] + **" "** + Tools.*toCharacterMonth*(Integer.*parseInt*(originalDate[1])) + **" "** + originalDate[2];  
 } **else if** (diffHours > 0)  
 **return "Last seen "** + diffHours + **" hours ago"**;  
 **else if** (diffMinutes > 0) {  
 **if** (diffMinutes <= 1) {  
 **return "Last seen 1 minute ago"**;  
 } **else** {  
 **return "Last seen "** + diffMinutes + **" minutes ago"**;  
 }  
 } **else return "Last seen a moment ago"**;  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 **return ""**;  
 }  
  
  
 }  
  
 **public static** String messageSentDateProper(String sentDate) {  
 String properDate = **""**;  
 Calendar cal = Calendar.*getInstance*();  
 Date todayDate = **new** Date();  
 cal.setTime(todayDate);  
 String[] date = sentDate.split(**" "**);  
 **int** todayMonth = cal.get(Calendar.***MONTH***) + 1;  
 **int** todayDay = cal.get(Calendar.***DAY\_OF\_MONTH***);  
 **if** (todayMonth == Integer.*parseInt*(date[1]) && todayDay == Integer.*parseInt*(date[0])) {  
 properDate = **"Today"** + **" "** + date[3] + **" "** + date[4];  
 *// 06 11 17 12:28 AM* } **else if** (todayMonth == Integer.*parseInt*(date[1]) && (todayDay - 1) == Integer.*parseInt*(date[0])) {  
 properDate = **"Yesterday"** + **" "** + date[3] + **" "** + date[4];  
 } **else** {  
 properDate = date[0] + **" "** + Tools.*toCharacterMonth*(Integer.*parseInt*(date[1])) + **" "** + date[2] + **" "** + date[3] + **" "** + date[4];  
 }  
 **return** properDate;  
 }  
  
 **public static boolean** isNetworkAvailable(Context context) {  
 ConnectivityManager connectivityManager  
 = (ConnectivityManager) context.getSystemService(Context.***CONNECTIVITY\_SERVICE***);  
 NetworkInfo activeNetworkInfo = connectivityManager.getActiveNetworkInfo();  
 **return** activeNetworkInfo != **null** && activeNetworkInfo.isConnected();  
 }  
}

RESOURCES : Menu

Menu\_main

*<?***xml version="1.0" encoding="utf-8"***?>*<**menu xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 tools:context="com.mys3soft.mys3chat.ActivityMain"**>  
  
 <**item  
 android:id="@+id/menu\_profile"  
 app:showAsAction="never"  
 android:title="@string/Profile"** />  
  
 <**item  
 android:id="@+id/menu\_addContacts"  
 android:title="@string/AddContacts"  
 app:showAsAction="never"** />  
  
 <**item  
 android:id="@+id/menu\_notification"  
 app:showAsAction="never"  
 android:title="Notifications"** />  
  
 <**item  
 android:id="@+id/menu\_refresh"  
 android:icon="@android:drawable/ic\_menu\_rotate"  
 app:showAsAction="always"  
 android:title="Refresh"** />  
  
 <**item  
 android:id="@+id/menu\_logout"  
 app:showAsAction="never"  
 android:title="@string/Logout"** />  
</**menu**>

Menu\_Chat

*<?***xml version="1.0" encoding="utf-8"***?>*<**menu  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"** >  
 <**item  
 android:id="@+id/menu\_phonecall"  
 app:showAsAction="always"  
 android:title="phone call"  
 android:icon="@android:drawable/ic\_menu\_call"** />  
 <**item  
 android:id="@+id/menu\_videocall"  
 app:showAsAction="always"  
 android:title="video call"  
 android:icon="@android:drawable/ic\_menu\_camera"** />  
 <**item  
 android:id="@+id/menu\_deleteConservation"  
 app:showAsAction="always"  
 android:title="Clear Conservation"  
 android:icon="@android:drawable/ic\_menu\_delete"** />  
  
 <**item  
 android:id="@+id/menu\_friendProfile"  
 android:title="Profile"  
 app:showAsAction="never"** />  
  
 <**item  
 android:id="@+id/menu\_deleteContact"  
 android:title="Delete Contact"  
 app:showAsAction="never"** />  
  
  
</**menu**>

RESOURCES : Layout

Activitiy\_main

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout  
 android:orientation="vertical"  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:id="@+id/main\_content"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:fitsSystemWindows="true"  
 tools:context="com.mys3soft.mys3chat.ActivityMain"**>  
  
 <**android.support.design.widget.AppBarLayout  
 android:id="@+id/appbar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:paddingTop="@dimen/appbar\_padding\_top"  
 android:theme="@style/AppTheme.AppBarOverlay"**>  
  
 <**android.support.v7.widget.Toolbar  
 android:id="@+id/toolbar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="?attr/actionBarSize"  
 android:background="?attr/colorPrimary"  
 app:layout\_scrollFlags="scroll|enterAlways"  
 app:popupTheme="@style/AppTheme.PopupOverlay"**>  
  
 </**android.support.v7.widget.Toolbar**>  
  
 <**android.support.design.widget.TabLayout  
 android:id="@+id/tabs"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 app:tabIndicatorColor="@color/tabsIndicatorColor"**/>  
  
 </**android.support.design.widget.AppBarLayout**>  
  
 <**android.support.v4.view.ViewPager  
 android:id="@+id/container"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 app:layout\_behavior="@string/appbar\_scrolling\_view\_behavior"** />  
  
  
</**LinearLayout**>

Activity\_login

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:gravity="center\_horizontal"  
 android:orientation="vertical"  
 tools:context="com.mys3soft.mys3chat.ActivityLogin"**>  
  
 <**ScrollView  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**>  
  
 <**LinearLayout  
 android:paddingTop="40dp"  
 android:layout\_marginLeft="20dp"  
 android:layout\_marginRight="20dp"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"**>  
 <**EditText  
 android:id="@+id/et\_Email"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="10dp"  
 android:hint="Email..."  
 android:inputType="textEmailAddress"  
 android:maxLines="1"** />  
  
 <**EditText  
 android:id="@+id/et\_Password"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="10dp"  
 android:hint="Password..."  
 android:inputType="textPassword"  
 android:maxLines="1"** />  
  
 <**Button  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/btn\_primary"  
 android:drawableLeft="@drawable/emoji\_1f464"  
 android:onClick="btnLoginClick"  
 android:paddingLeft="20dp"  
 android:paddingRight="20dp"  
 android:layout\_marginTop="40dp"  
 android:text="Login"  
 android:layout\_marginBottom="15dp"  
 android:textColor="#ffffff"** />  
  
 <**Button  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/btn\_success"  
 android:drawableLeft="@drawable/emoji\_1f465"  
 android:onClick="btnSignUpClick"  
 android:paddingLeft="20dp"  
 android:paddingRight="20dp"  
 android:text="Sign Up"  
 android:layout\_marginBottom="20dp"  
 android:textColor="#ffffff"** />  
 </**LinearLayout**>  
  
 </**ScrollView**>  
  
  
</**LinearLayout**>

Activity\_chat

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:emojicon="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 tools:context="com.mys3soft.mys3chat.ActivityChat"**>  
  
 <**android.support.design.widget.AppBarLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:paddingTop="@dimen/appbar\_padding\_top"  
 android:theme="@style/AppTheme.AppBarOverlay"**>  
  
 <**android.support.v7.widget.Toolbar  
 android:id="@+id/toolbarChatActivity"  
 android:layout\_width="match\_parent"  
 android:layout\_height="?attr/actionBarSize"  
 android:background="?attr/colorPrimary"  
 app:layout\_scrollFlags="scroll|enterAlways"  
 app:popupTheme="@style/AppTheme.PopupOverlay"**></**android.support.v7.widget.Toolbar**>  
 </**android.support.design.widget.AppBarLayout**>  
  
 <**android.support.v4.widget.SwipeRefreshLayout  
 android:id="@+id/swiperefresh"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_weight="10"**>  
  
 <**ScrollView  
 android:id="@+id/scrollView"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:paddingLeft="16dp"  
 android:paddingRight="16dp"  
 android:paddingTop="2dp"**>  
  
 <**LinearLayout  
 android:id="@+id/layout1"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"**>  
  
  
 *<!--<hani.momanii.supernova\_emoji\_library.Helper.EmojiconTextView-->  
 <!--android:layout\_width="wrap\_content"-->  
 <!--android:layout\_height="wrap\_content"-->  
 <!--android:layout\_centerHorizontal="true"-->  
 <!--android:layout\_centerVertical="true"-->  
 <!--android:layout\_marginTop="26dp"-->  
 <!--android:clickable="true"-->  
 <!--android:contextClickable="true"-->  
 <!--android:longClickable="true"-->  
 <!--android:text="Hello Emojis!"-->  
 <!--android:textAppearance="@style/TextAppearance.AppCompat.Large"-->  
 <!--android:textColor="#000000"-->  
 <!--emojicon:emojiconSize="45sp"-->  
 <!--emojicon:emojiconUseSystemDefault="true" />-->  
  
 <!--<TextView-->  
 <!--android:layout\_width="wrap\_content"-->  
 <!--android:layout\_height="wrap\_content"-->  
 <!--android:layout\_gravity="right"-->  
 <!--android:layout\_marginTop="10dp"-->  
 <!--android:layout\_weight="6"-->  
 <!--android:background="@drawable/messagebg1"-->  
 <!--android:padding="18dp"-->  
 <!--android:text="Test bottom"-->  
 <!--android:textColor="#000" />-->  
  
  
 <!--<TextView-->  
 <!--android:layout\_width="wrap\_content"-->  
 <!--android:layout\_height="wrap\_content"-->  
 <!--android:layout\_marginTop="10dp"-->  
 <!--android:layout\_weight="6"-->  
 <!--android:background="@drawable/messagebg2"-->  
 <!--android:gravity="left"-->  
 <!--android:padding="18dp"-->  
 <!--android:text="Test bottom"-->  
 <!--android:textColor="#000" />-->* </**LinearLayout**>  
  
 </**ScrollView**>  
  
  
 </**android.support.v4.widget.SwipeRefreshLayout**>  
  
 <**include  
 layout="@layout/message\_area"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:gravity="bottom"** />  
  
</**LinearLayout**>

Activity\_add\_contacts

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:padding="10dp"  
 tools:context="com.mys3soft.mys3chat.ActivityAddContact"**>  
  
  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center\_horizontal"  
 android:orientation="horizontal"**>  
 <**EditText  
 android:id="@+id/et\_SearchKey"  
 android:layout\_weight="1"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ems="10"  
 android:hint="Search Name , Email ..."  
 android:inputType="text"** />  
  
 <**ImageButton  
 android:id="@+id/bt\_Search"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:src="@drawable/emoji\_1f50d"  
 android:layout\_gravity="center"  
 android:background="@android:color/transparent"  
 android:onClick="btn\_SearchClick"**/>  
 *<!--<Button-->  
 <!--android:id="@+id/bt\_Search"-->  
 <!--android:layout\_width="wrap\_content"-->  
 <!--android:layout\_height="wrap\_content"-->  
 <!--android:onClick="btn\_SearchClick"-->  
 <!--android:text="Search" />-->* </**LinearLayout**>  
  
 <**ListView  
 android:id="@+id/lv\_AddContactList"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**>  
  
 </**ListView**>  
  
  
</**LinearLayout**>

Activity\_Friends\_profile

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:padding="20dp"  
 tools:context="com.mys3soft.mys3chat.ActivityFriendProfile"**>  
  
 <**ImageView  
 android:layout\_width="362dp"  
 android:layout\_height="wrap\_content"  
 android:src="@drawable/dp\_placeholder"** />  
  
 <**ImageButton  
 android:id="@+id/btn\_EditName2"  
 android:layout\_width="362dp"  
 android:layout\_height="wrap\_content"  
 android:background="@android:color/transparent"  
 android:onClick="btn\_EditNameClick"  
 android:src="@android:drawable/ic\_menu\_edit"** />  
  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/background\_round"  
 android:padding="10dp"  
 android:gravity="center"  
 android:orientation="horizontal"**>  
  
  
 <**TextView  
 android:id="@+id/tv\_FriendFullName\_L\_FriendProfile"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Full Name"  
 android:textColor="#ffffff"  
 android:layout\_weight="30"  
 android:textAlignment="center"  
 android:textSize="20dp"** />  
  
 <**ImageButton  
 android:id="@+id/btn\_EditName"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="@android:color/transparent"  
 android:onClick="btn\_EditNameClick"  
 android:src="@android:drawable/ic\_menu\_edit"**/>  
  
 </**LinearLayout**>  
  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="20"  
 android:gravity="center"  
 android:orientation="horizontal"**>  
  
 <**Button  
 android:id="@+id/btn\_AddFriend"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/btn\_primary"  
 android:drawableLeft="@drawable/ic\_menu\_invite"  
 android:drawableTint="#ffffff"  
 android:onClick="btn\_SendFriendRequestClick"  
 android:padding="4dp"  
 android:text="Add Contact"  
 android:textColor="#ffffff"** />  
 </**LinearLayout**>  
  
</**LinearLayout**>

Activity\_notifications

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 tools:context="com.mys3soft.mys3chat.ActivityNotifications"**>  
  
 <**TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_margin="10dp"  
 android:padding="4dp"  
 android:background="@drawable/btn\_primary"  
 android:text="Notifications"  
 android:textColor="#fff"  
 android:gravity="center"** />  
  
 <**ListView  
 android:id="@+id/lv\_NoticicationList"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**>  
  
 </**ListView**>  
  
</**LinearLayout**>

Activity\_Phonecall

*<?***xml version="1.0" encoding="utf-8"***?>*<**android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="#D3DFDD"  
 tools:context=".phonecall"**>  
  
 <**ImageView  
 android:id="@+id/imageView"  
 android:layout\_width="362dp"  
 android:layout\_height="wrap\_content"  
 android:src="@drawable/dp\_placeholder"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.498"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textView3"** />  
  
 <**TextView  
 android:id="@+id/textView2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="36dp"  
 android:width="200dp"  
 android:height="30dp"  
 android:text="@string/gadhewar\_saivarun"  
 android:textSize="14sp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.496"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 app:layout\_constraintVertical\_bias="0.02"** />  
  
 <**TextView  
 android:id="@+id/textView3"  
 android:layout\_width="71dp"  
 android:layout\_height="0dp"  
 android:layout\_marginTop="48dp"  
 android:text="@string/calling"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="@+id/textView2"** />  
  
 <**ImageButton  
 android:id="@+id/imageButton"  
 android:layout\_width="50dp"  
 android:layout\_height="50dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.499"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/imageView"  
 app:layout\_constraintVertical\_bias="0.458"  
 app:srcCompat="@drawable/endcall1"  
 android:contentDescription="TODO"** />  
  
</**android.support.constraint.ConstraintLayout**>

Activity\_profile

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:padding="10dp"  
 tools:context="com.mys3soft.mys3chat.ActivityProfile"**>  
  
 <**ImageView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:src="@drawable/dp\_placeholder"** />  
  
  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:background="@drawable/background\_round"  
 android:gravity="center"  
 android:orientation="horizontal"  
 android:padding="10dp"**>  
  
  
 <**TextView  
 android:id="@+id/tv\_UserFullName"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Full Name"  
 android:textColor="#ffffff"  
 android:textSize="20dp"** />  
  
  
 </**LinearLayout**>  
  
  
</**LinearLayout**>

Activity\_register

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:gravity="center"  
 android:orientation="vertical"  
 tools:context="com.mys3soft.mys3chat.ActivityRegister"**>  
  
 <**ScrollView  
 android:layout\_width="match\_parent"  
 android:layout\_gravity="center\_horizontal"  
 android:layout\_height="match\_parent"**>  
  
 <**LinearLayout  
 android:layout\_marginLeft="20dip"  
 android:layout\_marginRight="20dip"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"**>  
 <**EditText  
 android:id="@+id/et\_FirstName\_Rigister"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="10dip"  
 android:layout\_marginTop="50dip"  
 android:hint="Firstname..."  
 android:inputType="textCapSentences"  
 android:maxLines="1"** />  
 <**EditText  
 android:id="@+id/et\_LastName\_Rigister"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="10dip"  
 android:hint="Lastname..."  
 android:inputType="textCapSentences"  
 android:maxLines="1"** />  
 <**EditText  
 android:id="@+id/et\_Email\_Rigister"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="10dip"  
 android:hint="Email..."  
 android:inputType="textEmailAddress"  
 android:maxLines="1"** />  
 <**EditText  
 android:id="@+id/et\_Password\_Rigister"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="10dip"  
 android:hint="Password..."  
 android:inputType="textPassword"  
 android:maxLines="1"** />  
 <**Button  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="20dp"  
 android:layout\_marginTop="20dp"  
 android:paddingLeft="20dp"  
 android:paddingRight="20dp"  
 android:onClick="btn\_RegClick"  
 android:layout\_gravity="center"  
 android:background="@drawable/btn\_success"  
 android:drawableLeft="@drawable/emoji\_1f465"  
 android:text="Sign Up"  
 android:textColor="#ffffff"** />  
 </**LinearLayout**>  
  
 </**ScrollView**>  
  
</**LinearLayout**>

custom\_friend\_list\_row

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="horizontal"  
 android:padding="10dp"** >  
  
 <**TextView  
 android:id="@+id/tv\_HiddenEmail"  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:visibility="invisible"** />  
  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"**>  
  
 <**ImageView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:paddingRight="3dp"  
 android:src="@drawable/dp\_placeholder\_sm"** />  
  
 <**TextView  
 android:id="@+id/tv\_FriendFullName"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="41dp"  
 android:layout\_gravity="center\_vertical"  
 android:layout\_weight="1"  
 android:gravity="center\_vertical"  
 android:text="TextView"  
 android:textColor="#181717"  
 android:textSize="30px"** />  
  
  
 </**LinearLayout**>  
  
</**LinearLayout**>

Custom\_lastchat\_row

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:padding="10dp"**>  
  
  
 <**TextView  
 android:id="@+id/tv\_lastChat\_HiddenEmail"  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:visibility="invisible"** />  
  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"**>  
  
 <**ImageView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:src="@drawable/dp\_placeholder\_sm"  
 android:paddingRight="3dp"**/>  
  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"**>  
  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"**>  
  
 <**TextView  
 android:id="@+id/tv\_lastChat\_FriendFullName"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Full Name"  
 android:textColor="#181717"  
 android:textSize="25px"** />  
  
 <**TextView  
 android:id="@+id/tv\_lastChat\_MessageDate"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:gravity="right"  
 android:text="Sent Date"  
 android:textColor="#807474"  
 android:textSize="20px"** />  
  
 </**LinearLayout**>  
  
  
 <**hani.momanii.supernova\_emoji\_library.Helper.EmojiconTextView  
 android:id="@+id/tv\_lastChat\_Message"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Message..."  
 android:textColor="#807474"  
 android:textSize="20px"  
 app:emojiconSize="20px"** />  
  
 </**LinearLayout**>  
  
  
 </**LinearLayout**>  
  
</**LinearLayout**>

Custom\_notifications

*<?***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"  
 android:id="@+id/layout\_CustomNotificationRow"  
 android:padding="6dp"** >  
  
 <**ImageView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:src="@drawable/dp\_placeholder\_sm"  
 android:paddingRight="3dp"**/>  
  
  
 <**TextView  
 android:id="@+id/tv\_NotificationMessage"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Notification"  
 android:textSize="20dp"  
 android:textColor="#1e1d1d"  
 android:layout\_gravity="center"  
 android:paddingLeft="10dp"  
 android:layout\_weight="20"** />  
  
 *<!--<ImageButton-->  
 <!--android:layout\_width="wrap\_content"-->  
 <!--android:layout\_height="wrap\_content"-->  
 <!--android:layout\_gravity="center"-->  
 <!--android:background="@android:color/transparent"-->  
 <!--android:src="@drawable/emoji\_2705"/>-->  
  
 <!--<ImageButton-->  
 <!--android:layout\_width="wrap\_content"-->  
 <!--android:layout\_height="wrap\_content"-->  
 <!--android:layout\_gravity="center"-->  
 <!--android:background="@android:color/transparent"-->  
 <!--android:src="@drawable/emoji\_274c"/>-->*</**LinearLayout**>

Fragment\_chat

<**FrameLayout 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"**>  
  
 <**LinearLayout  
 android:orientation="vertical"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**>  
  
 <**ListView  
 android:id="@+id/lv\_LastChatList"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:longClickable="true"** >  
  
 </**ListView**>  
  
  
 </**LinearLayout**>  
  
  
</**FrameLayout**>

Fragment\_contact

<**FrameLayout 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"**>  
  
 <**LinearLayout  
 android:orientation="vertical"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**>  
  
 <**ListView  
 android:id="@+id/lv\_FriendList"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:longClickable="true"** >  
  
 </**ListView**>  
  
  
 </**LinearLayout**>  
  
</**FrameLayout**>

Message\_area

*<?***xml version="1.0" encoding="utf-8"***?>*<**LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:emojicon="http://schemas.android.com/apk/res-auto"  
 android:id="@+id/rootLayout"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:focusableInTouchMode="true"  
 android:gravity="bottom"  
 android:paddingRight="6dp"  
 android:paddingLeft="6dp"  
 android:paddingBottom="6dp"  
 android:orientation="horizontal"**>  
  
 <**ImageView  
 android:id="@+id/emoji\_btn"  
 android:layout\_width="35dp"  
 android:layout\_height="35dp"  
 android:paddingBottom="2dp"  
 android:src="@drawable/smiley"** />  
  
 <**hani.momanii.supernova\_emoji\_library.Helper.EmojiconEditText  
 android:id="@+id/et\_Message"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentBottom="true"  
 android:layout\_weight="1"  
 android:hint="Message..."  
 android:inputType="textMultiLine|textCapSentences"  
 android:maxHeight="80dp"  
 android:backgroundTint="@color/colorPrimary"  
 android:textColorHint="#CFD8DC"  
 emojicon:emojiconSize="28sp"** />  
  
 <**android.support.design.widget.FloatingActionButton  
 android:id="@+id/submit\_btn"  
 android:layout\_width="35dp"  
 android:layout\_height="35dp"  
 emojicon:fabSize="mini"  
 android:clickable="true"  
 android:onClick="btn\_SendMessageClick"  
 android:src="@android:drawable/ic\_menu\_send"  
 android:elevation="0dp"  
 android:background="#ffffff"  
 android:backgroundTintMode="src\_atop"  
 emojicon:pressedTranslationZ="0dp"** />  
  
</**LinearLayout**>

VALUES:

Colors

*<?***xml version="1.0" encoding="utf-8"***?>*<**resources**>  
 <**color name="colorPrimary"**>#006257</**color**>  
 <**color name="colorPrimaryDark"**>#1F6F61</**color**>  
 <**color name="colorAccent"**>#15695E</**color**>  
 <**color name="messageTextColor"**>#000000</**color**>  
 <**color name="tabsIndicatorColor"**>#ffffff</**color**>  
 <**color name="bg\_color"**>#ECF8F2</**color**>  
</**resources**>

Dimensions: dimens

<**resources**>  
 *<!-- Default screen margins, per the Android Design guidelines. -->* <**dimen name="activity\_horizontal\_margin"**>16dp</**dimen**>  
 <**dimen name="activity\_vertical\_margin"**>16dp</**dimen**>  
 <**dimen name="fab\_margin"**>16dp</**dimen**>  
 <**dimen name="appbar\_padding\_top"**>8dp</**dimen**>  
</**resources**>

Dimensions: dimens

<**resources**>  
 *<!-- Example customization of dimensions originally defined in res/values/dimens.xml  
 (such as screen margins) for screens with more than 820dp of available width. This  
 would include 7" and 10" devices in landscape (~960dp and ~1280dp respectively). -->* <**dimen name="activity\_horizontal\_margin"**>64dp</**dimen**>  
</**resources**>

Styles

<**resources**>  
  
 *<!-- Base application theme. -->* <**style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar"**>  
 *<!-- Customize your theme here. -->* <**item name="colorPrimary"**>@color/colorPrimary</**item**>  
 <**item name="colorPrimaryDark"**>@color/colorPrimaryDark</**item**>  
 <**item name="colorAccent"**>@color/colorPrimary</**item**>  
 <**item name="colorControlActivated"**>@color/colorPrimary</**item**>  
 <**item name="android:windowBackground"**>@color/bg\_color</**item**>  
 </**style**>  
  
 <**style name="AppTheme.NoActionBar"**>  
 <**item name="windowActionBar"**>false</**item**>  
 <**item name="windowNoTitle"**>true</**item**>  
 </**style**>  
  
 <**style name="AppTheme.AppBarOverlay" parent="ThemeOverlay.AppCompat.Dark.ActionBar"** >  
  
 </**style**>  
  
 <**style name="AppTheme.PopupOverlay" parent="ThemeOverlay.AppCompat.Light"**>  
  
 </**style**>  
  
 *<!--<style name="GlobalTheme" parent="@android:style/Theme.Light">-->  
 <!--<item name="android:windowBackground">@color/bg\_color</item>-->  
 <!--</style>-->*</**resources**>

Strings

<**resources**>  
 <**string name="app\_name"**>MyS3Chat</**string**>  
 <**string name="Logout"**>Logout</**string**>  
 <**string name="Profile"**>Profile</**string**>  
 <**string name="Contacts"**>Contacts</**string**>  
 <**string name="AddContacts"**>Add Contacts</**string**>  
 <**string name="title\_activity\_main"**>MyS3Chat</**string**>  
 <**string name="action\_settings"**>Settings</**string**>  
 <**string name="section\_format"**>Hello World from section: %1$d</**string**>  
  
 *<!--* ***TODO: Remove or change this placeholder text*** *-->* <**string name="hello\_blank\_fragment"**>Hello blank fragment</**string**>  
 <**string name="gadhewar\_saivarun\_string"**>Gadhewar Saivarun . . . .@string</**string**>  
 <**string name="calling"**>Calling ...</**string**>  
 <**string name="gadhewar\_saivarun"**>Gadhewar Saivarun....</**string**>  
</**resources**>