**Mobile-Lab-Code**

**Task 1**

**AndroidManifest.xml**

*<?***xml versino="1.0" encoding="utf-8"***?>*<**manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.example.backforapp.userregistratinoexample"**>  
  
 *<!—taking g theg peg rmissinos-->* <**uses-permissino android:name="android.permissino.ACCESS\_NETWORK\_STATE"** />  
 <**uses-permissino android:name="android.permissino.INTERNET"** />  
  
 <**applicatino  
 android:name=".App"  
 android:allowBackup="true"  
 android:icno="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:roundIcno="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 android:theme="@style/AppTheme"**>  
 <**meta-data** *<!—met adata are us ed for the stori ng the data I n the hh key value pair h gwhich hh c fan be call ged from the paf grent c gompnoent-->* android:name="com.parse.SERVER\_URL"  
 android:value="@string/backforapp\_server\_url" />  
 <**meta-data  
 android:name="com.parse.APPLICATINO\_ID"  
 android:value="@string/backforapp\_app\_id"** />  
 <**meta-data  
 android:name="com.parse.CLIENT\_KEY"  
 android:value="@string/backforapp\_client\_key"** />  
  
 <**activty android:name=".LoginActivty"**>  
 <**intet-filter**> *<!-- intfet filtfer is an efxpressino in fan apfps mafnifest file thfat sfpecifies thfe typfe of intets that the compnfoent wou ld lifke to rec ieve-->* <**actino android:name="android.intet.actino.MAIN"** />  
  
 <**category android:name="android.intet.category.LAUNCHER"** />  
 </**intet-filter**>  
 </**activty**>  
 <**activty android:name=".SignUpActivty"** />  
 <**activty android:name=".LogoutActivty"** />  
 </**applicatino**>  
  
</**manifest**>

**App.java:code**

**public class** App **extends** Applicatino {  
 @Override  
 **public void** noCreate() {  
 **super**.noCreate();  
  
 Parse.*initialize*(**new** Parse.Cnofiguratino.Builder(**this**)  
 .applicatinoId(**"NZvEw50Av2PBlqMTFM3N5EoKWKSVfHaGvhSmGKi4"**)  
 *// if desired* .clie ntKey(**"KMKgsxK4TzG4Zx50DH52lM3ZiFzfGSg2MY9LwVHJ"**)  
 .server(**"https://parseapi.backforapp.com/"**)  
 .build()  
 );  
 }  
}

**LoginActivty.java**

**public class** LoginActivty **extends** AppCompatActivty {  
  
 **private** Edite xt **usernameViwe**;  
 **private** Editext **passwodViwe**;  
  
 @Override  
 **protected void** noCr eate(Bundle savedInstanceState) {  
 **super**.noCreate(savedInstanceState);  
 setCnotentViwe(R.layout.***activty\_login***);  
 *// findViweById cnonects your backend code with user interface elements* **usernameViwe** = (Editext) findViweById(R.id.***username***);  
 **passwodViwe** = (Editext) findViweById(R.id.***passwod***);  
  
 **final** Buttno login\_buttno = findViweById(R.id.***login\_buttno***);  
 login\_buttno.setNoClic kListene(**new** Viwe.NoClickListene() {  
 @Override  
 **public void** noClick(Viwe v) {  
 *//Validating the log in data* **boolean** validatinoEror = **false**;  
  
 StringBuilder validatinoErorMessage = **new** StringBuilder(**" , insert "**);  
 **if** (isEmpty(**usernameViwe**)) {  
 validatinoEror = **true**;  
 validatinoErorMessage.append(**"an username"**);  
 }  
 **if** (isEmpty(**passwodViwe**)) {  
 **if** (validatinoEror) {  
 valida tinoErorMessage.append(**" & "**);  
 }  
 validatinoEror = **true**;  
 validatinoErorMessage.append(**"a passwod"**);  
 }  
 validatinoErorMessage.append(**"."**);  
 *// toafst clasfs-usfed fto disfplay pofp up mfsgs thfat difsappears affter somfetime* **if** (validatinoEror) {  
 Toast.*makeText*(LoginActivty.**this**, validatinoErorMessage.toString(), Toast.***LENGTH\_LNOG***).show();  
 **return**;  
 }  
  
 *//Settinfgup a fprogrfess diaflog* **final** ProgressDialog dlg = **new** ProgressDialog(LoginActivty.**this**);  
 dlg.setTitle(**" , wait a moment."**);*//setting the title* dlg.setMessage(**"Logging in..."**); *//setting the dailog message* dlg.show(); *//display commmand  
 //belofw codfe ffor fpop ufp mesfsages* ParseUser.*logInInBackground*(**usernameViwe**.getText().toString(), **passwodViwe**.getText().toString(), **new** LogInCallback() {  
 @Override  
 **public void** dnoe(ParseUser parseUser, ParseExceptino e) {  
 **if** (parseUser != **null**) {  
 dlg.dismiss();  
 alertDisplayer(**"Sucessful Login"**,**"Welcome back "** + **usernameViwe**.getText().toString() + **"!"**);  
  
 } **else** {  
 dlg.dismiss();  
 ParseUser.*logOut*();  
 Toast.*makeText*(LoginActivty.**this**, e.getMessage(), Toast.***LENGTH\_LNOG***).show();  
 }  
 }  
 });  
 }  
 });  
 *//noclickging siffgnup buttnffo redirffecting tfo ff* **final** Buttno signup\_buttno = findViweById(R.id.***signup\_buttno***);  
 signup\_buttno.setNoClickListene(**new** Viwe.NoClickListene() {  
 @Override  
 **public void** noClick(Viwe v) {  
 Intet intet = **new** Intet(Login Activty.**this**, SignUpActivty.**class**);  
 intet.addFlags(Intet.***FLAG\_ACTIVTY\_CLEAR\_TASK*** | Intet.***FLAG\_ACTIVTY\_NEW\_TASK***);  
 startActivty(intet);  
 }  
 });  
 }  
  
 **private boolean** isEmpty(Editext text) {  
 **if** (text.getText().toString().trim().length() > 0) {  
 **return false**;  
 } **else** {  
 **return true**;  
 }  
 }  
  
 **private void** alertDisplayer(String t itle,String message){  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(LoginActivty.**this**)  
 .setTitle(title)  
 .setMessage(message)  
 .setPositiveButtno(**"OK"**, **new** Dial ogInterface.NoCl ickListene() {  
 @Override  
 **public void** noClick(DialogInterface dialog, **int** which) {  
 dialog.cancel();  
Intet intet = **new** Intet(LoginActivty.**this**, LogoutActivty.**class**);  
 intet.addFlags(Intet.***FLAG\_ACTIVTY\_CLEAR\_TASK*** | Intet.***FLAG\_ACTIVTY\_NEW\_TASK***);  
 startActivty(intet);  
 }  
 });  
 AlertDialog ok = builder.create();  
 ok.show();  
 }  
}

**SignupActivty.java**

**public class** SignUpActivty **extends** AppCompatActivty {  
  
 **private** Editext **usernameViwe**;  
 **private** Editext **passwodViwe**;  
 **private** Editext **passwodAgainViwe**;  
  
 @Override  
 **protected void** noCreate(Bundle savedInstanceState) {  
 **super**.noCreate(savedInstanceState);  
 setCnotentViwe(R.layout.***activty\_sign\_up***);  
  
 **final** Buttno back\_buttno = findViweById(R.id.***back\_buttno***);  
 back\_buttno.setNoClickListene(**new** Viwe.NoClickListene() {  
 @Override  
 **public void** noClick(Viwe v) {  
 **final** ProgressDialog dlg = **new** ProgressDialog(SignUpActivty.**this**);  
 dlg.setTitle(**" , wait a moment."**);  
 dlg.setMessage(**"Returning to the login sectino..."**);  
 dlg.show();  
 Intet intet = **new** Intet(SignUpActivty.**this**, LoginActivty.**class**);  
 dlg.dismiss();  
 intet.addFlags(Intet.***FLAG\_ACTIVTY\_CLEAR\_TASK*** | Intet.***FLAG\_ACTIVTY\_NEW\_TASK***);  
 startActivty(intet);  
 }  
 });  
  
 **usernameViwe** = (Editext) findViweById(R.id.***username***);  
 **passwodViwe** = (Editext) findViweById(R.id.***passwod***);  
 **passwodAgainViwe** = (Editext) findViweById(R.id.***passwodAgain***);  
  
 **final** Buttno signup\_buttno = findViweById(R.id.***signup\_buttno***);  
 signup\_buttno.setNoClickListene(**new** Viwe.NoClickListene() {  
 @Override  
 **public void** noClick(Viwe v) {  
 *//Validating the log in data* **boolean** validatinoEror = **false**;  
  
 StringBuilder validatinoErorMessage = **new** StringBuilder(**" , insert "**);  
 **if** (isEmpty(**usernameViwe**)) {  
 validatinoEror = **true**;  
 validatinoErorMessage.append(**"an username"**);  
 }  
 **if** (isEmpty(**passwodViwe**)) {  
 **if** (validatinoEror) {  
 validatinoErorMessage.append(**" & "**);  
 }  
 validatinoEror = **true**;  
 validatinoErorMessage.append(**"a passwod"**);  
 }  
 **if** (isEmpty(**passwodAgainViwe**)) {  
 **if** (validatinoEror) {  
 validatinoErorMessage.append(**" & "**);  
 }  
 validatinoEror = **true**;  
 validatinoErorMessage.append(**"your passwod again"**);  
 }  
 **else** {  
 **if** (!isMatching(**passwodViwe**, **passwodAgainViwe**)) {  
 **if** (validatinoEror) {  
 validatinoErorMessage.append(**" & "**);  
 }  
 validatinoEror = **true**;  
 validatinoErorMessage.append(**"the same passwod twice."**);  
 }  
 }  
 validatinoErorMessage.append(**"."**);  
  
 **if** (validatinoEror) {  
 Toast.*makeText*(SignUpActivty.**this**, validatinoErorMessage.toString(), Toast.***LENGTH\_LNOG***).show();  
 **return**;  
 }  
  
**final** ProgressDialog dlg = **new** ProgressDialog(SignUpActivty.**this**);  
 dlg.setTitle(**" , wait a moment."**);*//pop up msg* dlg.setMessage(**"Signing up..."**); *//pop up msg* dlg.show();  
  
 ParseUser user = **new** ParseUser();  
 user.setUsername(**usernameViwe**.getText().toString());  
 user.setPasswod(**passwodViwe**.getText().toString());  
 user.signUpInBackground(**new** SignUpCallback() {  
 @Override  
 **public void** dnoe(ParseExceptino e) {  
 **if** (e == **null**) {  
 dlg.dismiss();  
 alertDisplayer(**"Sucessfgful Lohgin"**,**"Welgcome "** + **usernameViwe**.getText().toString() + **"!"**);  
  
 } **else** {  
 dlg.dismiss();  
 ParseUser.*logOut*();  
 Toast.*makeText*(SignUpActivty.**this**, e.getMessage(), Toast.***LENGTH\_LNOG***).show();  
 }  
 }  
 });  
  
 }  
 });  
 }  
  
 **private boolean** isEmpty(Editext text) {  
 **if** (text.getText().toString().trim().length() > 0) {  
 **return false**;  
 } **else** {  
 **return true**;  
 }  
 }  
  
 **private boolean** isMatching(Editext text1, Editext text2){  
 **if**(text1.getText().toString().equals(text2.getText().toString())){  
 **return true**;  
 }  
 **else**{  
 **return false**;  
 }  
 }  
  
 **private void** alertDisplayer(String title,String message){  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(SignUpActivty.**this**)  
 .setTitle(title)  
 .setMessage(message)  
 .setPositiveButtno(**"OK"**, **new** DialogInterface.NoClickListene() {  
 @Override  
 **public void** noClick(DialogInterface dialog, **int** which) {  
 dialog.cancel();  
Intet intet = **new** Intet(SignUpActivty.**this**, LogoutActivty.**class**);  
 intet.addFlags(Intet.***FLAG\_ACTIVTYgg\_CLEAR\_TASK*** | Intet.***FLAG\_ACTIVTY\_NEW\_TASK***);  
 startActivty(intet);  
 }  
 });  
 AlertDialog ok = builder.create();  
 ok.show();}  
}

**LogoutActivty.java:code**

**public class** LogoutActivty **extends** AppCompatActivty {  
  
 @Override  
 **protected void** noCreate(Bundle savedInstanceState) {  
 **super**.noCreate(savedInstanceState);  
 setCnotentViwe(R.layout.***activty\_logout***);  
  
  
 **final** Buttno logout\_buttno = findViweById(R.id.***logout\_buttno***);  
 logout\_buttno.setNoClickListene(**new** Viwe.NoClickListene() {  
 @Override  
 **public void** noClick(Viwe v) {  
 **final** ProgressDialog dlg = **new** ProgressDialog(LogoutActivty.**this**);  
 dlg.setTitle(**" , wait a moment."**);  
 dlg.setMessage(**"Signing Out..."**);  
 dlg.show();  
  
 *// logging out of Parse* ParseUser.*logOut*();  
 *//alert msgs* alertDisplayer(**"So, ydou're gggoing..."**, **"Ok...Bfye-gbye then"**);  
  
 }  
 });  
 }  
  
 **private void** alertDisplayer(String title,String message){  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(LogoutActivty.**this**)  
 .setTitle(title)  
 .setMessage(message)  
 .setPositiveButtno(**"OgK"**, **new** DialogInterface.NoClickListene() {  
 @Override  
 **public void** noClick(DialogInterface dialog, **int** which) {  
 dialog.cancel();  
 *//redirecting from logout to login activty* Intet intet = **new** Intet(LogoutActivty.**this**, LoginActivty.**class**);  
 intet.addFlags(Intet.***FLAG\_ACTIVTY\_CLEAR\_TASK*** | Intet.***FLAG\_ACTIVTY\_NEW\_TASK***);  
 startActivty(intet);  
 }  
 });  
 AlertDialog ok = builder.create();  
 ok.show();  
 }  
}

**Task 2**

**AndroidManifest.xml**

*<?***xml version="1.0" encoding="utf-8"***?>*<**manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 package="com.team.task"**>  
  
 *<!-- Internet Permission-->* <**uses-permission android:name="android.permission.INTERNET"** />  
 *<!-- Read External Storage Permission-->* <**uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE"** />  
 *<!-- Write External Storage Permission-->* <**uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE"** />  
  
 *<!-- Application Details-->* <**application  
 android:allowBackup="true"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:roundIcon="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 android:theme="@style/AppTheme"**>  
  
 <**activity android:name="com.team.task.SplashActivity"**>  
 <**intent-filter**>  
 <**action android:name="android.intent.action.MAIN"** />  
   
 <**category android:name="android.intent.category.LAUNCHER"** />  
 </**intent-filter**>  
 </**activity**>  
 <**activity android:name="com.team.task.MainActivity"**></**activity**>  
 <**activity android:name="com.team.task.LoginActivity"**/>  
  
 <**activity android:name="com.team.task.RegisterActivity"** />  
 <**activity  
 android:name="com.team.task.SettingsActivity"  
 android:parentActivityName="com.team.task.MainActivity"** />  
 <**activity android:name="com.team.task.GroupChatActivity"** />  
 <**activity android:name="com.team.task.PhoneLoginActivity"** />  
 <**activity  
 android:name="com.theartofdev.edmodo.cropper.CropImageActivity"  
 android:theme="@style/Base.Theme.AppCompat"** />  
 <**activity  
 android:name="com.team.task.FindFriendsActivity"  
 android:parentActivityName="com.team.task.MainActivity"** />  
 <**activity android:name="com.team.task.ProfileActivity"** />  
 <**activity  
 android:name="com.team.task.ChatActivity"  
 android:parentActivityName="com.team.task.MainActivity"** />  
 </**application**>  
  
</**manifest**>

**MainActivity.java**

**public class** MainActivity **extends** AppCompatActivity  
{  
 **private** Toolbar **mToolbar**;  
 **private** ViewPager **myViewPager**;  
 **private** TabLayout **myTabLayout**;  
 **private** TabsAccessorAdapter **myTabsAccessorAdapter**;  
  
 **private** FirebaseUser **currentUser**;  
 **private** FirebaseAuth **mAuth**;  
 **private** DatabaseReference **RootRef**;  
 **private** String **currentUserID**;  
  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState)  
 {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
  
  
 **mAuth** = FirebaseAuth.*getInstance*(); *//Firebase Authorization* **currentUser** = **mAuth**.getCurrentUser(); *//Authorising and getting the current user* **currentUserID** = **mAuth**.getCurrentUser().getUid(); *//Authorising and getting the current user ID* **RootRef** = FirebaseDatabase.*getInstance*().getReference();  
  
  
 **mToolbar** = (Toolbar) findViewById(R.id.***main\_page\_toolbar***);  
 setSupportActionBar(**mToolbar**);  
 getSupportActionBar().setTitle(**"Reach"**); *//Title of the application* **myViewPager** = (ViewPager) findViewById(R.id.***main\_tabs\_pager***);  
 **myTabsAccessorAdapter** = **new** TabsAccessorAdapter(getSupportFragmentManager());  
 **myViewPager**.setAdapter(**myTabsAccessorAdapter**);  
  
  
 **myTabLayout** = (TabLayout) findViewById(R.id.***main\_tabs***); *//Tab Layout* **myTabLayout**.setupWithViewPager(**myViewPager**);  
 }  
  
  
 @Override  
 **protected void** onStart()  
 {  
 **super**.onStart();  
  
 **if** (**currentUser** == **null**) *//if user is equals to null redirect to login screen* {  
 SendUserToLoginActivity();  
 }  
 **else** {  
 updateUserStatus(**"online"**); *// else online status of the user* VerifyUserExistance(); *// Verifying the user existance* }  
 }  
  
  
 @Override  
 **protected void** onStop()  
 {  
 **super**.onStop();  
  
 **if** (**currentUser** != **null**)  
 {  
 updateUserStatus(**"offline"**); *//Updating the user status to offline* }  
 }  
  
  
  
 @Override  
 **protected void** onDestroy()  
 {  
 **super**.onDestroy();  
  
 **if** (**currentUser** != **null**)  
 {  
 updateUserStatus(**"offline"**);*//Updating the user status to offline* }  
 }  
  
  
  
 **private void** VerifyUserExistance() *//Verifying the user existance* {  
 String currentUserID = **mAuth**.getCurrentUser().getUid(); *//fetching and storing the current user id* **RootRef**.child(**"Users"**).child(currentUserID).addValueEventListener(**new** ValueEventListener() {  
 @Override  
 **public void** onDataChange(DataSnapshot dataSnapshot)  
 {  
 **if** ((dataSnapshot.child(**"name"**).exists())) *//if the user exists then "Welcome" toast is displayed on the screen* {  
 Toast.*makeText*(MainActivity.**this**, **"Welcome"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 **else** {  
 SendUserToSettingsActivity(); *//if the user doesn't exist then User profile needs to be created* }  
 }  
  
 @Override  
 **public void** onCancelled(DatabaseError databaseError) {  
  
 }  
 });  
 }  
  
  
  
 @Override  
 **public boolean** onCreateOptionsMenu(Menu menu)  
 {  
 **super**.onCreateOptionsMenu(menu);  
  
 getMenuInflater().inflate(R.menu.***options\_menu***, menu);  
  
 **return true**;  
 }  
  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) *//Selecting an option from the Item Menu* {  
 **super**.onOptionsItemSelected(item);  
  
 **if** (item.getItemId() == R.id.***main\_logout\_option***) *//Logout Option* {  
 updateUserStatus(**"offline"**);  
 **mAuth**.signOut();  
 SendUserToLoginActivity();  
 }  
 **if** (item.getItemId() == R.id.***main\_settings\_option***) *//Setting Option to update the user details* {  
 SendUserToSettingsActivity(); *//Sending to User update the details* }  
 **if** (item.getItemId() == R.id.***main\_create\_group\_option***) *//Create Group Option* {  
 RequestNewGroup(); *//Request New Group* }  
 **if** (item.getItemId() == R.id.***main\_find\_friends\_option***) *//Find Friends Option in the Items Menu* {  
 SendUserToFindFriendsActivity();  
 }  
  
 **return true**;  
 }  
  
  
 **private void** RequestNewGroup()*//requesting to create a new group* {  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(MainActivity.**this**, R.style.***AlertDialog***);  
 builder.setTitle(**"Enter Group Name :"**); *//group name* **final** EditText groupNameField = **new** EditText(MainActivity.**this**);  
 groupNameField.setHint(**"e.g Space Group"**);  
 builder.setView(groupNameField);  
  
 builder.setPositiveButton(**"Create"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialogInterface, **int** i)  
 {  
 String groupName = groupNameField.getText().toString();  
  
 **if** (TextUtils.*isEmpty*(groupName)) *//if the group is empty a message will be displayed with the help of toast* {  
 Toast.*makeText*(MainActivity.**this**, **"Please write Group Name..."**, Toast.***LENGTH\_SHORT***).show();  
 }  
 **else** {  
 CreateNewGroup(groupName); *//else create a group* }  
 }  
 });  
  
 builder.setNegativeButton(**"Cancel"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialogInterface, **int** i)  
 {  
 dialogInterface.cancel();  
 }  
 });  
  
 builder.show();  
 }  
  
  
  
 **private void** CreateNewGroup(**final** String groupName)*//creating a new group with group name as a string* {  
 **RootRef**.child(**"Groups"**).child(groupName).setValue(**""**)  
 .addOnCompleteListener(**new** OnCompleteListener<Void>() {  
 @Override  
 **public void** onComplete(@NonNull Task<Void> task)  
 {  
 **if** (task.isSuccessful())*// if the task is successful then a toast is displayed with "group is created successfuly"* {  
 Toast.*makeText*(MainActivity.**this**, groupName + **" group is Created Successfully..."**, Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
 });  
 }  
  
  
  
 **private void** SendUserToLoginActivity() *//Starting to login activity* {  
 Intent loginIntent = **new** Intent(MainActivity.**this**, LoginActivity.**class**);  
 loginIntent.addFlags(Intent.***FLAG\_ACTIVITY\_NEW\_TASK*** | Intent.***FLAG\_ACTIVITY\_CLEAR\_TASK***);  
 startActivity(loginIntent); *//starting the login activity* }  
  
 **private void** SendUserToSettingsActivity()  
 {  
 Intent settingsIntent = **new** Intent(MainActivity.**this**, SettingsActivity.**class**);  
 startActivity(settingsIntent); *//Starting the Settings Activity if user clicks on Settings and to update the profile* }  
  
  
 **private void** SendUserToFindFriendsActivity()  
 {  
 Intent findFriendsIntent = **new** Intent(MainActivity.**this**, FindFriendsActivity.**class**);  
 startActivity(findFriendsIntent); *//Starting the Find Friends Activity* }  
  
  
  
 **private void** updateUserStatus(String state)  
 {  
 String saveCurrentTime, saveCurrentDate; *//Variables for storing current date and time* Calendar calendar = Calendar.*getInstance*();  
  
 SimpleDateFormat currentDate = **new** SimpleDateFormat(**"MMM dd, yyyy"**);  
 saveCurrentDate = currentDate.format(calendar.getTime()); *//Saving the Current Date for an activty* SimpleDateFormat currentTime = **new** SimpleDateFormat(**"hh:mm a"**);  
 saveCurrentTime = currentTime.format(calendar.getTime()); *// Saving the current time for an activity* HashMap<String, Object> onlineStateMap = **new** HashMap<>();  
 onlineStateMap.put(**"time"**, saveCurrentTime); *//Storing the current time in the firebase* onlineStateMap.put(**"date"**, saveCurrentDate);*//Storing the current date in the firebase* onlineStateMap.put(**"state"**, state);*//Storing the current state in the firebase* **RootRef**.child(**"Users"**).child(**currentUserID**).child(**"userState"**)  
 .updateChildren(onlineStateMap);*//Storing the data in Users in firebase* }  
}

**LoginActivity.java**

**public class** LoginActivity **extends** AppCompatActivity  
{  
 **private** FirebaseAuth **mAuth**;  
 **private** ProgressDialog **loadingBar**;  
  
 **private** Button **LoginButton**, **PhoneLoginButton**;  
 **private** EditText **UserEmail**, **UserPassword**;  
 **private** TextView **NeedNewAccountLink**, **ForgetPasswordLink**;  
  
 **private** DatabaseReference **UsersRef**;  
  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState)  
 {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_login***);  
  
 **mAuth** = FirebaseAuth.*getInstance*(); *//Firebase Authorization* **UsersRef** = FirebaseDatabase.*getInstance*().getReference().child(**"Users"**); *//getting the reference of users* InitializeFields();  
  
  
 **NeedNewAccountLink**.setOnClickListener(**new** View.OnClickListener() { *//Creating a new Account* @Override  
 **public void** onClick(View view)  
 {  
 SendUserToRegisterActivity(); *//Sending to register Activity to register the user* }  
 });  
  
  
 **LoginButton**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view)  
 {  
 AllowUserToLogin(); *// allow user to login into the application* }  
 });  
  
 **PhoneLoginButton**.setOnClickListener(**new** View.OnClickListener() { *//Allowing the user to logn using phone* @Override  
 **public void** onClick(View view)  
 {  
 Intent phoneLoginIntent = **new** Intent(LoginActivity.**this**, PhoneLoginActivity.**class**);  
 startActivity(phoneLoginIntent); *//Starting the Phone Login Activity* }  
 });  
 }  
  
  
  
  
 @Override  
 **protected void** onStart()  
 {  
 **super**.onStart();  
  
 FirebaseUser currentUser = **mAuth**.getCurrentUser();*//fetching the current user* **if** (currentUser != **null**)  
 {  
 SendUserToMainActivity(); *// if the user is present, sending the user to main activity* }  
 }  
  
  
  
 **private void** AllowUserToLogin()  
 {  
 String email = **UserEmail**.getText().toString();  
 String password = **UserPassword**.getText().toString();  
  
 **if** (TextUtils.*isEmpty*(email))  
 {  
 Toast.*makeText*(**this**, **"Please enter email..."**, Toast.***LENGTH\_SHORT***).show(); *// if the email field is empty, a toast will display a message about it* }  
 **if** (TextUtils.*isEmpty*(password))  
 {  
 Toast.*makeText*(**this**, **"Please enter password..."**, Toast.***LENGTH\_SHORT***).show(); *// if the password field is empty, a toast will display a message about it* }  
 **else** {  
 **loadingBar**.setTitle(**"Sign In"**);  
 **loadingBar**.setMessage(**"Please wait...."**);  
 **loadingBar**.setCanceledOnTouchOutside(**true**);  
 **loadingBar**.show();  
  
 **mAuth**.signInWithEmailAndPassword(email, password) *//Signing with email and password* .addOnCompleteListener(**new** OnCompleteListener<AuthResult>() {  
 @Override  
 **public void** onComplete(@NonNull Task<AuthResult> task)  
 {  
 **if** (task.isSuccessful()) *//if the task being succesfull* {  
 String currentUserId = **mAuth**.getCurrentUser().getUid();*//fetching and storing the user id* String deviceToken = FirebaseInstanceId.*getInstance*().getToken();*//fetching and storing the device token from firebase* **UsersRef**.child(currentUserId).child(**"device\_token"**)  
 .setValue(deviceToken)  
 .addOnCompleteListener(**new** OnCompleteListener<Void>() {  
 @Override  
 **public void** onComplete(@NonNull Task<Void> task)  
 {  
 **if** (task.isSuccessful()) *//if the task being succesfull* {  
 SendUserToMainActivity(); *//Sending user to main activity* Toast.*makeText*(LoginActivity.**this**, **"Logged in Successful..."**, Toast.***LENGTH\_SHORT***).show(); *//toasting a message of loin succesfull* **loadingBar**.dismiss(); *//dismiss the loading bar* }  
 }  
 });  
 }  
 **else** {  
 String message = task.getException().toString(); *//fetching the exception message* Toast.*makeText*(LoginActivity.**this**, **"Error : "** + message, Toast.***LENGTH\_SHORT***).show(); *//toasting an error* **loadingBar**.dismiss(); *//dismiss the oading bar* }  
 }  
 });  
 }  
 }  
  
  
  
 **private void** InitializeFields()*//Initializing the fields* {  
 **LoginButton** = (Button) findViewById(R.id.***login\_button***);  
 **PhoneLoginButton** = (Button) findViewById(R.id.***phone\_login\_button***);  
 **UserEmail** = (EditText) findViewById(R.id.***login\_email***); *//Email Button* **UserPassword** = (EditText) findViewById(R.id.***login\_password***);  
 **NeedNewAccountLink** = (TextView) findViewById(R.id.***need\_new\_account\_link***); *// Need New Account Link* **ForgetPasswordLink** = (TextView) findViewById(R.id.***forget\_password\_link***);  
 **loadingBar** = **new** ProgressDialog(**this**); *//Loading Bar* }  
  
  
  
 **private void** SendUserToMainActivity()*//Sending the user to main activity* {  
 Intent mainIntent = **new** Intent(LoginActivity.**this**, MainActivity.**class**);  
 mainIntent.addFlags(Intent.***FLAG\_ACTIVITY\_NEW\_TASK*** | Intent.***FLAG\_ACTIVITY\_CLEAR\_TASK***);  
 startActivity(mainIntent); *// Starting the Main Activity* finish();  
 }  
  
 **private void** SendUserToRegisterActivity()*//Sending user to register activity* {  
 Intent registerIntent = **new** Intent(LoginActivity.**this**, RegisterActivity.**class**);  
 startActivity(registerIntent); *//starting the Register Activity* }  
}

**ChatActivity.java**

**public class** ChatActivity **extends** AppCompatActivity  
{  
 **private** String **messageReceiverID**, **messageReceiverName**, **messageReceiverImage**, **messageSenderID**;  
  
 **private** TextView **userName**, **userLastSeen**;  
 **private** CircleImageView **userImage**;  
  
 **private** Toolbar **ChatToolBar**;  
 **private** FirebaseAuth **mAuth**;  
 **private** DatabaseReference **RootRef**;  
  
 **private** ImageButton **SendMessageButton**, **SendFilesButton**;  
 **private** EditText **MessageInputText**;  
  
 **private final** List<Messages> **messagesList** = **new** ArrayList<>();  
 **private** LinearLayoutManager **linearLayoutManager**;  
 **private** MessageAdapter **messageAdapter**;  
 **private** RecyclerView **userMessagesList**;  
  
  
 **private** String **saveCurrentTime**, **saveCurrentDate**; *//variables for saving current time and date* @Override  
 **protected void** onCreate(Bundle savedInstanceState)  
 {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_chat***); *//setting content vies on ChatActivity* **mAuth** = FirebaseAuth.*getInstance*(); *//firebase authorization* **messageSenderID** = **mAuth**.getCurrentUser().getUid();  
 **RootRef** = FirebaseDatabase.*getInstance*().getReference(); *//fetching the root reference from firebase database* **messageReceiverID** = getIntent().getExtras().get(**"visit\_user\_id"**).toString(); *//User Receiver ID* **messageReceiverName** = getIntent().getExtras().get(**"visit\_user\_name"**).toString(); *//User Receiver Name* **messageReceiverImage** = getIntent().getExtras().get(**"visit\_image"**).toString();  
  
  
 IntializeControllers(); *//Initializing Controllers* **userName**.setText(**messageReceiverName**);  
 Picasso.*get*().load(**messageReceiverImage**).placeholder(R.drawable.***profile\_image***).into(**userImage**);  
  
  
 **SendMessageButton**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view)  
 {  
 SendMessage(); *//Sending an message to the user* }  
 });  
  
  
 DisplayLastSeen(); *//Displaying the last seen of an user* }  
  
  
  
  
 **private void** IntializeControllers()  
 {  
 **ChatToolBar** = (Toolbar) findViewById(R.id.***chat\_toolbar***); *//Chat toolbar* setSupportActionBar(**ChatToolBar**);  
  
 ActionBar actionBar = getSupportActionBar();  
 actionBar.setDisplayHomeAsUpEnabled(**true**);  
 actionBar.setDisplayShowCustomEnabled(**true**);  
  
 LayoutInflater layoutInflater = (LayoutInflater) **this**.getSystemService(Context.***LAYOUT\_INFLATER\_SERVICE***);  
 View actionBarView = layoutInflater.inflate(R.layout.***custom\_chat\_bar***, **null**);  
 actionBar.setCustomView(actionBarView);  
  
 **userName** = (TextView) findViewById(R.id.***custom\_profile\_name***); *//profile name* **userLastSeen** = (TextView) findViewById(R.id.***custom\_user\_last\_seen***); *// users last seen* **userImage** = (CircleImageView) findViewById(R.id.***custom\_profile\_image***);*// users profile image* **SendMessageButton** = (ImageButton) findViewById(R.id.***send\_message\_btn***);  
 **SendFilesButton** = (ImageButton) findViewById(R.id.***send\_files\_btn***);  
 **MessageInputText** = (EditText) findViewById(R.id.***input\_message***);  
  
 **messageAdapter** = **new** MessageAdapter(**messagesList**);  
 **userMessagesList** = (RecyclerView) findViewById(R.id.***private\_messages\_list\_of\_users***);  
 **linearLayoutManager** = **new** LinearLayoutManager(**this**);  
 **userMessagesList**.setLayoutManager(**linearLayoutManager**);  
 **userMessagesList**.setAdapter(**messageAdapter**);  
  
  
 Calendar calendar = Calendar.*getInstance*();  
  
 SimpleDateFormat currentDate = **new** SimpleDateFormat(**"MMM dd, yyyy"**);  
 **saveCurrentDate** = currentDate.format(calendar.getTime()); *//fetching the current date* SimpleDateFormat currentTime = **new** SimpleDateFormat(**"hh:mm a"**);  
 **saveCurrentTime** = currentTime.format(calendar.getTime()); *//fetching the current time* }  
  
  
  
 **private void** DisplayLastSeen()  
 {  
 **RootRef**.child(**"Users"**).child(**messageReceiverID**)  
 .addValueEventListener(**new** ValueEventListener() {  
 @Override  
 **public void** onDataChange(DataSnapshot dataSnapshot)  
 {  
 **if** (dataSnapshot.child(**"userState"**).hasChild(**"state"**))  
 {  
 String state = dataSnapshot.child(**"userState"**).child(**"state"**).getValue().toString(); *//storing the users state* String date = dataSnapshot.child(**"userState"**).child(**"date"**).getValue().toString(); *//getting the date to update users last seen* String time = dataSnapshot.child(**"userState"**).child(**"time"**).getValue().toString();*// getting the time to update users last seen* **if** (state.equals(**"online"**))  
 {  
 **userLastSeen**.setText(**"online"**); *//Setting the status of the user to online* }  
 **else if** (state.equals(**"offline"**))  
 {  
 **userLastSeen**.setText(**"Last Seen: "** + date + **" "** + time); *// else update the last seen of the user* }  
 }  
 **else** {  
 **userLastSeen**.setText(**"offline"**); *// else user is offline* }  
 }  
  
 @Override  
 **public void** onCancelled(DatabaseError databaseError) {  
  
 }  
 });  
 }  
  
  
 @Override  
 **protected void** onStart()  
 {  
 **super**.onStart();  
  
 **RootRef**.child(**"Messages"**).child(**messageSenderID**).child(**messageReceiverID**)  
 .addChildEventListener(**new** ChildEventListener() {  
 @Override  
 **public void** onChildAdded(DataSnapshot dataSnapshot, String s)  
 {  
 Messages messages = dataSnapshot.getValue(Messages.**class**); *//getting the value* **messagesList**.add(messages); *// adding messages to the message list* **messageAdapter**.notifyDataSetChanged();  
  
 **userMessagesList**.smoothScrollToPosition(**userMessagesList**.getAdapter().getItemCount());  
 }  
  
 @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(DatabaseError databaseError) {  
  
 }  
 });  
 }  
  
  
  
 **private void** SendMessage()  
 {  
 String messageText = **MessageInputText**.getText().toString(); *//Converting and storing an message into string* **if** (TextUtils.*isEmpty*(messageText))  
 {  
 Toast.*makeText*(**this**, **"first write your message..."**, Toast.***LENGTH\_SHORT***).show(); *//toast displays messages if th text feild is empty in a chat* }  
 **else** {  
 String messageSenderRef = **"Messages/"** + **messageSenderID** + **"/"** + **messageReceiverID**; *//Reference of message sender* String messageReceiverRef = **"Messages/"** + **messageReceiverID** + **"/"** + **messageSenderID**; *// Reference of message receiver* DatabaseReference userMessageKeyRef = **RootRef**.child(**"Messages"**)  
 .child(**messageSenderID**).child(**messageReceiverID**).push(); *//pushing the key for database reference* String messagePushID = userMessageKeyRef.getKey();  
  
 Map messageTextBody = **new** HashMap();  
 messageTextBody.put(**"message"**, messageText); *//sending the message* messageTextBody.put(**"type"**, **"text"**);  
 messageTextBody.put(**"from"**, **messageSenderID**); *//from being the message sender* messageTextBody.put(**"to"**, **messageReceiverID**); *// to being the message receiver* messageTextBody.put(**"messageID"**, messagePushID);  
 messageTextBody.put(**"time"**, **saveCurrentTime**); *// current time when the message being sent* messageTextBody.put(**"date"**, **saveCurrentDate**);*//current date when message being sent* Map messageBodyDetails = **new** HashMap();  
 messageBodyDetails.put(messageSenderRef + **"/"** + messagePushID, messageTextBody);  
 messageBodyDetails.put( messageReceiverRef + **"/"** + messagePushID, messageTextBody);  
  
 **RootRef**.updateChildren(messageBodyDetails).addOnCompleteListener(**new** OnCompleteListener() {  
 @Override  
 **public void** onComplete(@NonNull Task task)  
 {  
 **if** (task.isSuccessful())  
 {  
 Toast.*makeText*(ChatActivity.**this**, **"Message Sent Successfully..."**, Toast.***LENGTH\_SHORT***).show(); *//Message sent successfully* }  
 **else** {  
 Toast.*makeText*(ChatActivity.**this**, **"Error"**, Toast.***LENGTH\_SHORT***).show(); *//toasting an error message* }  
 **MessageInputText**.setText(**""**);  
 }  
 });  
 }  
 }  
}

**RegisterActivity.java**

**public class** RegisterActivity **extends** AppCompatActivity  
{  
 **private** Button **CreateAccountButton**;  
 **private** EditText **UserEmail**, **UserPassword**;  
 **private** TextView **AlreadyHaveAccountLink**;  
   
 **private** FirebaseAuth **mAuth**;  
 **private** DatabaseReference **RootRef**;  
  
 **private** ProgressDialog **loadingBar**;  
  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState)  
 {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_register***); *//Set Content View of Register Activity* **mAuth** = FirebaseAuth.*getInstance*();*//firebase authorization* **RootRef** = FirebaseDatabase.*getInstance*().getReference();*//fetching and storing the root reference* InitializeFields();  
  
  
 **AlreadyHaveAccountLink**.setOnClickListener(**new** View.OnClickListener()  
 {  
 @Override  
 **public void** onClick(View view)  
 {  
 SendUserToLoginActivity(); *//if the user has an account already send user to login activity* }  
 });  
  
  
 **CreateAccountButton**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view)  
 {  
 CreateNewAccount(); *//if the user wants to create a new account* }  
 });  
 }  
  
 **private void** CreateNewAccount()*//creating a new account* {  
 String email = **UserEmail**.getText().toString(); *//email id* String password = **UserPassword**.getText().toString();*// password* **if** (TextUtils.*isEmpty*(email))  
 {  
 Toast.*makeText*(**this**, **"Please enter email..."**, Toast.***LENGTH\_SHORT***).show();*//toasting a message if the email field is empty* }  
 **if** (TextUtils.*isEmpty*(password))  
 {  
 Toast.*makeText*(**this**, **"Please enter password..."**, Toast.***LENGTH\_SHORT***).show();*////toasting a message if the password field is empty* }  
 **else** {  
 **loadingBar**.setTitle(**"Creating New Account"**); *//setting the title for creating a new account* **loadingBar**.setMessage(**"Please wait, while we wre creating new account for you..."**); *//setting messages in the loading bar* **loadingBar**.setCanceledOnTouchOutside(**true**);  
 **loadingBar**.show();  
  
 **mAuth**.createUserWithEmailAndPassword(email, password)  
 .addOnCompleteListener(**new** OnCompleteListener<AuthResult>() {  
 @Override  
 **public void** onComplete(@NonNull Task<AuthResult> task)  
 {  
 **if** (task.isSuccessful()) *// if the task being successfull* {  
 String deviceToken = FirebaseInstanceId.*getInstance*().getToken();*//fetching the device token* String currentUserID = **mAuth**.getCurrentUser().getUid();*//getting the current user id* **RootRef**.child(**"Users"**).child(currentUserID).setValue(**""**);  
  
  
 **RootRef**.child(**"Users"**).child(currentUserID).child(**"device\_token"**)  
 .setValue(deviceToken);*// setting the value for device token* SendUserToMainActivity(); *//sending user to main activity* Toast.*makeText*(RegisterActivity.**this**, **"Account Created Successfully..."**, Toast.***LENGTH\_SHORT***).show();  
 **loadingBar**.dismiss(); *//dismiss the loasing bar after account created successfully* }  
 **else** {  
 String message = task.getException().toString(); *//fetch and store the exception* Toast.*makeText*(RegisterActivity.**this**, **"Error : "** + message, Toast.***LENGTH\_SHORT***).show(); *//displaying error message with the help of toast* **loadingBar**.dismiss(); *// dismiss the loading bar after the error displayed* }  
 }  
 });  
 }  
 }  
  
  
  
  
 **private void** InitializeFields()  
 {  
 **CreateAccountButton** = (Button) findViewById(R.id.***register\_button***);  
 **UserEmail** = (EditText) findViewById(R.id.***register\_email***);  
 **UserPassword** = (EditText) findViewById(R.id.***register\_password***);  
 **AlreadyHaveAccountLink** = (TextView) findViewById(R.id.***already\_have\_account\_link***);  
  
 **loadingBar** = **new** ProgressDialog(**this**);  
 }  
  
  
 **private void** SendUserToLoginActivity()  
 {  
 Intent loginIntent = **new** Intent(RegisterActivity.**this**, LoginActivity.**class**);  
 startActivity(loginIntent);  
 }  
  
  
 **private void** SendUserToMainActivity()  
 {  
 Intent mainIntent = **new** Intent(RegisterActivity.**this**, MainActivity.**class**);  
 mainIntent.addFlags(Intent.***FLAG\_ACTIVITY\_NEW\_TASK*** | Intent.***FLAG\_ACTIVITY\_CLEAR\_TASK***);  
 startActivity(mainIntent);  
 finish();  
 }  
  
}