**Firebase Login Activity :**

public class MainActivity extends AppCompatActivity {

**Variable Declaration**

public static final String USER\_ID = "trailblazelearn.nus.edu.sg.trailblazelearn.userid";

private Spinner training\_mod, spinner2;

SignInButton Button;

DatabaseReference db;

FirebaseAuth mAuth;

private CallbackManager mCallbackManager;

List<Account> accounts;

List<String> emailval;

private Session session;//global variable

private ProgressDialog progressDialog;

private final static int RC\_SIGN\_IN=2;

GoogleApiClient mGoogleApiClient;

FirebaseAuth.AuthStateListener mAuthListener;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

FacebookSdk.sdkInitialize(getApplicationContext());

setContentView(R.layout.activity\_main);

training\_mod = (Spinner) findViewById(R.id.trainingmode);

SignInButton signInButton = findViewById(R.id.sign\_in\_button);

// Session Manager

session = new Session(getApplicationContext());

**Sign in button click for firebase**

signInButton.setOnClickListener(new OnClickListener() {

@Override

onClick(View v) {

signIn(); }

});

db = FirebaseDatabase.getInstance().getReference("Account");

mAuth = FirebaseAuth.getInstance();

initializeFB();

mAuthListener=new FirebaseAuth.AuthStateListener() {

@Override

public void onAuthStateChanged(@NonNull final FirebaseAuth firebaseAuth) {

if(firebaseAuth.getCurrentUser() !=null){

FirebaseUser user=firebaseAuth.getCurrentUser();

final String email = user.getEmail();

db.addListenerForSingleValueEvent(new ValueEventListener() {

@Override

public void onDataChange(DataSnapshot dataSnapshot) {

FirebaseUser username=firebaseAuth.getCurrentUser();

String str = username.getDisplayName();

String[] splitStr = str.split("\\s+");

String accounttype = "Google";

}

});

}else{

}

}

};

**Requesting Token**

GoogleSignInOptions gso = new GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT\_SIGN\_IN)

.requestIdToken(getString(R.string.default\_web\_client\_id))

.requestEmail()

.build();

// Build a GoogleSignInClient with the options specified by gso.

mGoogleApiClient = new GoogleApiClient.Builder(this)

.enableAutoManage(this /\* FragmentActivity \*/,

new GoogleApiClient.OnConnectionFailedListener() {

@Override

public void onConnectionFailed(@NonNull ConnectionResult connectionResult) {

Toast.makeText(MainActivity.this, "Something went wrong", Toast.LENGTH\_SHORT).show();

}

})

.addApi(Auth.GOOGLE\_SIGN\_IN\_API,gso)

.build();

}

**Initialize FB**

private void initializeFB() {

// Initialize Facebook Login button

mCallbackManager = CallbackManager.Factory.create();

LoginButton loginButton = findViewById(R.id.button\_facebook\_login);

loginButton.setReadPermissions("email", "public\_profile");

loginButton.registerCallback(mCallbackManager, new FacebookCallback<LoginResult>() {

@Override

public void onSuccess(LoginResult loginResult) {

handleFacebookAccessToken(loginResult.getAccessToken());

}

@Override

public void onCancel() {

}

@Override

public void onError(FacebookException error) {

}

});

}

// [START auth\_with\_facebook]

private void handleFacebookAccessToken(AccessToken token) {

AuthCredential credential = FacebookAuthProvider.getCredential(token.getToken());

mAuth.signInWithCredential(credential)

.addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {

@Override

public void onComplete(@NonNull Task<AuthResult> task) {

if (task.isSuccessful()) {

// Sign in success, update UI with the signed-in user's information

FirebaseUser user = mAuth.getCurrentUser();

moveToNextActivity();

Toast.makeText(MainActivity.this, user.getDisplayName(), Toast.LENGTH\_SHORT).show();

//updateUI(user);

} else {

// If sign in fails, display a message to the user.

Toast.makeText(MainActivity.this, "Authentication failed or account exist with same email ID but different provider.",

Toast.LENGTH\_SHORT).show();

}

}

});

}

**Move to next activity**

private void moveToNextActivity() {

Intent i = null;

if(training\_mod.getSelectedItem().equals("Trainers")){

i = new Intent(MainActivity.this, LearningTrailActivity.class);

}else{

i = new Intent(MainActivity.this, TrailStationActivity.class);

}

startActivity(i);

}

**Sign Out**

public void signOut() {

mAuth.signOut();

LoginManager.getInstance().logOut();

}

private void signIn() {

progressDialog = new ProgressDialog(MainActivity.this,

R.style.AppTheme\_Dark\_Dialog);

progressDialog.setIndeterminate(true);

progressDialog.setMessage("Authenticating...");

progressDialog.show();

Intent signInIntent = Auth.GoogleSignInApi.getSignInIntent(mGoogleApiClient);

startActivityForResult(signInIntent, RC\_SIGN\_IN);

}

**// Configure Google Sign In**

@Override

public void onActivityResult(int requestCode, int resultCode, Intent data) {

super.onActivityResult(requestCode, resultCode, data);

// Result returned from launching the Intent from GoogleSignInClient.getSignInIntent(...);

if (requestCode == RC\_SIGN\_IN) {

// The Task returned from this call is always completed, no need to attach

// a listener.

GoogleSignInResult result=Auth.GoogleSignInApi.getSignInResultFromIntent(data);

if(result.isSuccess()) {

GoogleSignInAccount account=result.getSignInAccount();

firebaseAuthWithGoogle(account);

}else{

Toast.makeText(MainActivity.this, "Auth went wrong", Toast.LENGTH\_SHORT).show();

}

}else {

mCallbackManager.onActivityResult(requestCode, resultCode, data);

}

}

**// Auth Google Sign In**

private void firebaseAuthWithGoogle(GoogleSignInAccount account) {

AuthCredential credential = GoogleAuthProvider.getCredential(account.getIdToken(), null);

mAuth.signInWithCredential(credential)

.addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {

@Override

public void onComplete(@NonNull Task<AuthResult> task) {

if (task.isSuccessful()) {

// Sign in success, update UI with the signed-in user's information

Log.d("TAG", "signInWithCredential:success");

FirebaseUser user = mAuth.getCurrentUser();

String useridval=user.getUid();

String usernamval=user.getDisplayName();

session.createLoginSession(usernamval, useridval);

//Remove after implementing check with db

// Intent i = new Intent(getApplicationContext(), LearningTrailActivity.class);

// startActivity(i);

moveToNextActivity();

progressDialog.dismiss();

Toast.makeText(MainActivity.this, user.getDisplayName(), Toast.LENGTH\_SHORT).show();

// updateUI(user);

} else {

// If sign in fails, display a message to the user.

Log.w("TAG", "signInWithCredential:failure", task.getException());

Toast.makeText(MainActivity.this, "Auth went wrong", Toast.LENGTH\_SHORT).show();

// updateUI(null);

}

// ...

}

});

}

private boolean checkuserexist(String email, DataSnapshot dataSnapshot){

for (DataSnapshot messageSnapshot: dataSnapshot.getChildren()) {

String name = (String) messageSnapshot.child("emailaddress").getValue();

String message = (String) messageSnapshot.child("firstname").getValue();

if(name==email){

return true;

}

// emailval.add(name);

}

return false;

}

private void addUser(String firstname,String lastname,String email,String accounttype) {

//checking if the value is provided

if (!TextUtils.isEmpty(email)) {

//getting a unique id using push().getKey() method

//it will create a unique id and we will use it as the Primary Key for our Artist

String id = db.push().getKey();

//creating an Artist Object

Account account = new Account(id, firstname, lastname,email,accounttype,true);

//Saving the Artist

db.child(id).setValue(account);

} else {

//if the value is not given displaying a toast

Toast.makeText(this, "User not able to register", Toast.LENGTH\_LONG).show();

}

}

}

**Learning Trail Activity :**

public class LearningTrailActivity extends AppCompatActivity{

private boolean mTwoPane;

private RecyclerView recyclerView;

private LearningTrailAdapter mAdapter;

DatabaseReference db;

ManageLearningTrail trailhelper;

// Session Manager Class

Session session;

private ArrayList<LearningTrial> learningTrails = new ArrayList<>();

private String userId;

SwipeController swipeController = null;

FirebaseAuth mAuth;

FirebaseAuth.AuthStateListener mAuthListener;

@Override

protected void onStart() {

super.onStart();

mAuth.addAuthStateListener(mAuthListener);

db.addValueEventListener(new ValueEventListener() {

@Override

public void onDataChange(DataSnapshot dataSnapshot) {

learningTrails.clear();

for (DataSnapshot ds : dataSnapshot.getChildren()) {

LearningTrial learningTrial = ds.getValue(LearningTrial.class);

if (learningTrial.getUserid().equals(userId)) {

learningTrails.add(learningTrial);

}

}

mAdapter = new LearningTrailAdapter(LearningTrailActivity.this, learningTrails);

recyclerView.setAdapter(mAdapter);

}

@Override

public void onCancelled(DatabaseError databaseError) {

}

});

}

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_learning\_trail);

Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);

setSupportActionBar(toolbar);

FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);

fab.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

//Snackbar.make(view, "Replace with your own action", Snackbar.LENGTH\_LONG)

//.setAction("Action", null).show();

Intent i = new Intent(LearningTrailActivity.this, AddLearningTrailActivity.class);

startActivity(i);

// displayInputDialog();

}

});

mAuth = FirebaseAuth.getInstance();

userId = mAuth.getCurrentUser().getUid();

mAuthListener = new FirebaseAuth.AuthStateListener() {

@Override

public void onAuthStateChanged(@NonNull FirebaseAuth firebaseAuth) {

if (firebaseAuth.getCurrentUser() == null) {

Intent i = new Intent(LearningTrailActivity.this, MainActivity.class);

startActivity(i);

} else {

userId = mAuth.getCurrentUser().getUid();

}

}

};

if (findViewById(R.id.module\_detail\_container) != null) {

// The detail container view will be present only in the

// large-screen layouts (res/values-w900dp).

// If this view is present, then the

// activity should be in two-pane mode.

mTwoPane = true;

}

recyclerView = findViewById(R.id.module\_list);

recyclerView.setLayoutManager(new LinearLayoutManager(this));

//INITIALIZE FIREBASE DB

db= FirebaseDatabase.getInstance().getReference("LearningTrail");

trailhelper=new ManageLearningTrail(db);

//ADAPTER

mAdapter=new LearningTrailAdapter(LearningTrailActivity.this,trailhelper.retrieve());

recyclerView.setAdapter(mAdapter);

//adding swipe function

swipeController = new SwipeController(new SwipeControllerActions() {

@Override

public void onRightClicked(int position) {

//mAdapter.players.remove(position);

//learningTrails.remove(position);

deleteTrail(position);

mAdapter.notifyItemRemoved(position);

mAdapter.notifyItemRangeChanged(position, mAdapter.getItemCount());

}

@Override

public void onLeftClicked(int position) {

//Redirect to Edit activity with data.

Intent i = new Intent(LearningTrailActivity.this, EditLearningTrailActivity.class);

i.putExtra("LearningTrailId", learningTrails.get(position).getLearningtrailid());

startActivity(i);

}

});

ItemTouchHelper itemTouchhelper = new ItemTouchHelper(swipeController);

itemTouchhelper.attachToRecyclerView(recyclerView);

recyclerView.addItemDecoration(new RecyclerView.ItemDecoration() {

@Override

public void onDraw(Canvas c, RecyclerView parent, RecyclerView.State state) {

swipeController.onDraw(c);

}

});

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.main\_menu, menu);

menu.findItem(R.id.action\_trainer).setVisible(false);

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

// Handle action bar item clicks here. The action bar will

// automatically handle clicks on the Home/Up button, so long

// as you specify a parent activity in AndroidManifest.xml.

int id = item.getItemId();

switch (item.getItemId()) {

case R.id.action\_settings:

Toast.makeText(LearningTrailActivity.this, "For Future use.", Toast.LENGTH\_SHORT).show();

return true;

case R.id.action\_participant:

Intent i = new Intent(LearningTrailActivity.this, TrailStationActivity.class);

startActivity(i);

return true;

case R.id.action\_logout:

logoutUser();

return true;

default:

return super.onOptionsItemSelected(item);

}

}

protected void logoutUser() {

FirebaseAuth.getInstance().signOut();

mAuth = FirebaseAuth.getInstance();

mAuthListener=new FirebaseAuth.AuthStateListener() {

@Override

public void onAuthStateChanged(@NonNull FirebaseAuth firebaseAuth) {

if(firebaseAuth.getCurrentUser() ==null){

// Intent i = new Intent(LearningTrailActivity.this, MainActivity.class);

//startActivity(i);

session.logoutUser();

}else{

}

}

};

}

public void deleteTrail(int position){

LearningTrial lt = learningTrails.get(position);

db.child(lt.getLearningtrailid()).setValue(null);

}

}

**Learning Trail Adapter Class:**

public class LearningTrailAdapter extends RecyclerView.Adapter<LearningTrailAdapter.MyViewHolder> {

public Context context;

public LearningTrailActivity mParentActivity;

public List<LearningTrial> mValues;

public boolean mTwoPane;

public static final String LEARNING\_TRAIL\_ID = "trailblazelearn.nus.edu.sg.trailblazelearn.learningtrailid";

public static final String TRAIL\_NAME = "trailblazelearn.nus.edu.sg.trailblazelearn.trailname";

public LearningTrailAdapter(Context parent, List<LearningTrial> cartList) {

this.context = parent;

this.mValues = cartList;

}

@Override

public void onBindViewHolder(MyViewHolder holder, int position) {

holder.learningtrailid.setText(mValues.get(position).getLearningtrailid());

holder.trailname.setText(mValues.get(position).getTrailname());

holder.thumbnail.setImageResource(R.drawable.wall4);

holder.itemView.setTag(mValues.get(position));

holder.setItemClickListener(new ItemClickListener() {

@Override

public void onItemClick(int pos) {

//OPEN DETAI ACTIVITY

openDetailActivity(mValues.get(pos).getLearningtrailid(),mValues.get(pos).getTrailname(),mValues.get(pos).getUserid());

}

});

}

@Override

public MyViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {

View view = LayoutInflater.from(parent.getContext())

.inflate(R.layout.module\_list\_content, parent, false);

return new MyViewHolder(view);

}

public class MyViewHolder extends RecyclerView.ViewHolder implements View.OnClickListener {

public TextView learningtrailid,userid,trailname;

public Date traildate;

public ImageView thumbnail;

public RelativeLayout viewBackground, viewForeground;

ItemClickListener itemClickListener;

public MyViewHolder(View view) {

super(view);

learningtrailid = (TextView) view.findViewById(R.id.learningtrailid);

trailname = (TextView) view.findViewById(R.id.learningtrailname);

// userid = (TextView) view.findViewById(R.id.userid);

thumbnail = view.findViewById(R.id.thumbnail);

viewBackground = view.findViewById(R.id.view\_background);

viewForeground = view.findViewById(R.id.view\_foreground);

itemView.setOnClickListener(this);

}

public void setItemClickListener(ItemClickListener itemClickListener)

{

this.itemClickListener=itemClickListener;

}

public void onClick(View view) {

this.itemClickListener.onItemClick(this.getLayoutPosition());

}

}

@Override

public int getItemCount() {

return mValues.size();

}

private void openDetailActivity(String...details)

{

Intent i=new Intent(context,LearningTrailDetailActivity.class);

i.putExtra("LEARNING\_TRAIL\_ID",details[0]);

i.putExtra("LEARNING\_TRAIL\_NAME",details[1]);

i.putExtra("USER\_ID",details[2]);

context.startActivity(i);

}

}

**Data Access Layer :**

public class ManageLearningTrail {

DatabaseReference db;

Boolean saved=null;

ArrayList<LearningTrial> learningTrials=new ArrayList<>();

private String userId;

public ManageLearningTrail(DatabaseReference db) {

this.db = db;

}

//WRITE IF NOT NULL

public Boolean save(LearningTrial learningTrial)

{

if(learningTrial==null)

{

saved=false;

}else

{

try

{

db.push().setValue(learningTrial);

saved=true;

}catch (DatabaseException e)

{

e.printStackTrace();

saved=false;

}

}

return saved;

}

//IMPLEMENT FETCH DATA AND FILL ARRAYLIST

private void fetchData(DataSnapshot dataSnapshot)

{

learningTrials.clear();

for (DataSnapshot ds : dataSnapshot.getChildren())

{

String uId = String.valueOf(ds.child("userid").getValue());

if(userId != null && uId.equals(userId)) {

LearningTrial learningTrial = ds.getValue(LearningTrial.class);

learningTrials.add(learningTrial);

}

}

}

//READ THEN RETURN ARRAYLIST

public ArrayList<LearningTrial> retrieve() {

db.addChildEventListener(new ChildEventListener() {

@Override

public void onChildAdded(DataSnapshot dataSnapshot, String s) {

fetchData(dataSnapshot);

}

@Override

public void onChildChanged(DataSnapshot dataSnapshot, String s) {

fetchData(dataSnapshot);

}

@Override

public void onChildRemoved(DataSnapshot dataSnapshot) {

}

@Override

public void onChildMoved(DataSnapshot dataSnapshot, String s) {

}

@Override

public void onCancelled(DatabaseError databaseError) {

}

});

return learningTrials;

}

public boolean remove(int position){

LearningTrial lt = learningTrials.get(position);

db.child(lt.getLearningtrailid()).setValue(null);

return true;

}

public String getTrailId(int position){

LearningTrial lt = learningTrials.get(position);

return lt.getLearningtrailid();

}

//READ THEN RETURN ARRAYLIST

public LearningTrial retrieveTrailbyId(String TrailId) {

//Learning trail object and return to UI so that can get data.

return null;

}

public void setUserId(String userId) {

this.userId = userId;

}

}