Submitted By: Zahid Bin Amin

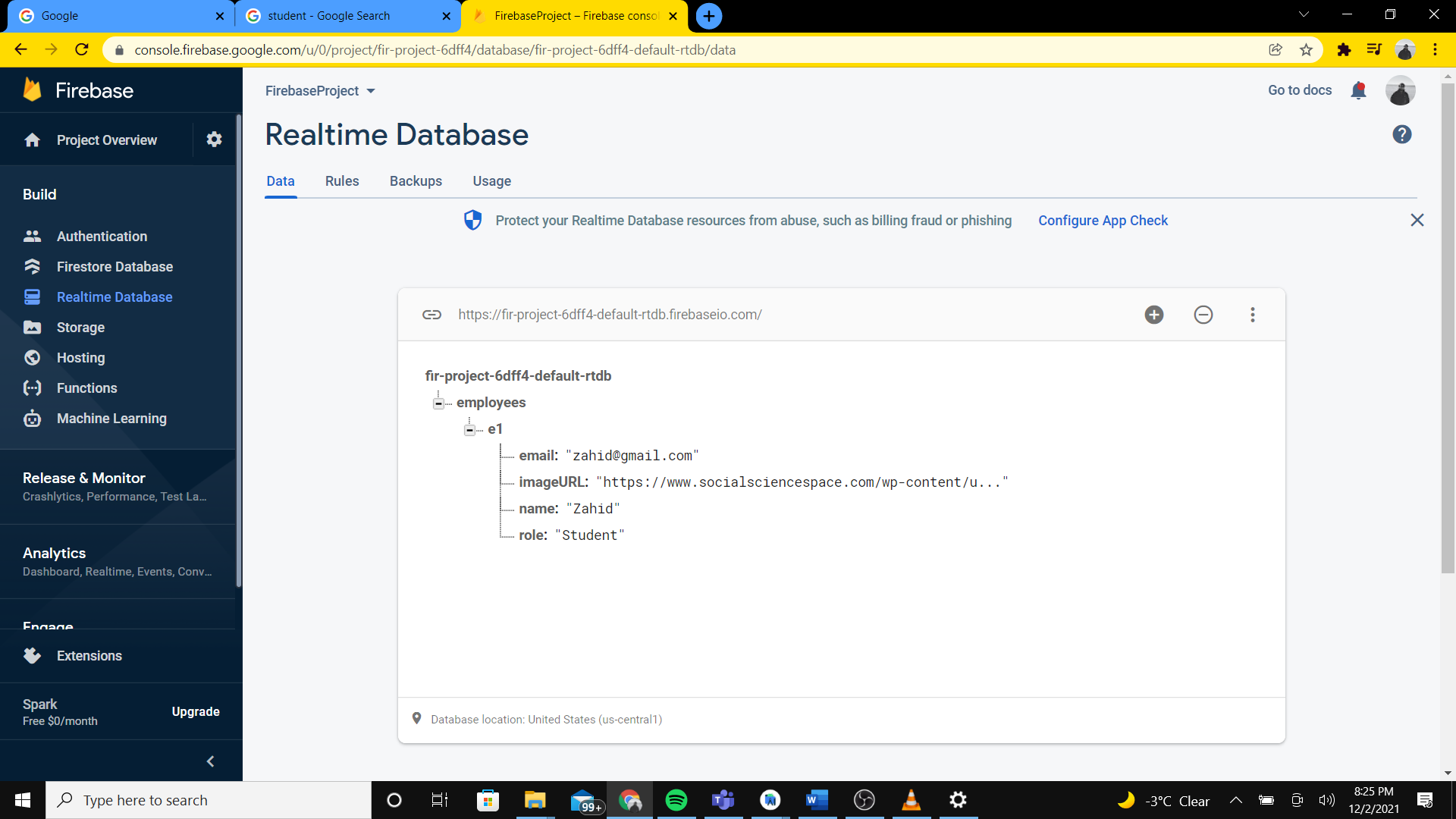
Date: 12/2/2021

**Tutorial to read and add data to Firebase Realtime Database in Android Studio**

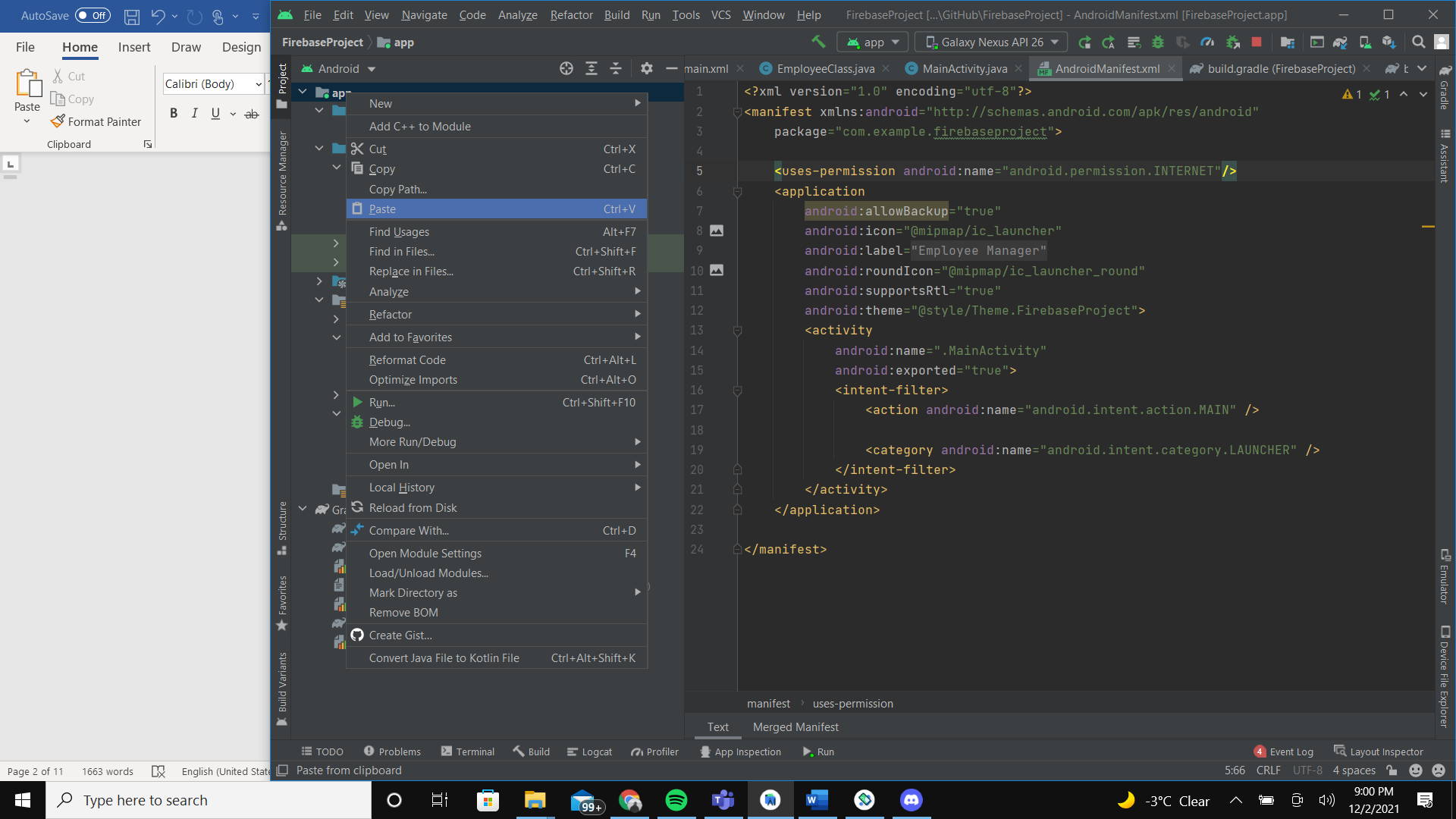
**Click the link to watch my walkthrough of the tutorial if you are not interested in reading the following long paragraphs.**

[**https://drive.google.com/file/d/1y0EicuWWsbWIaUm4gU-t\_KL96JsZIlqv/view?usp=sharing**](https://drive.google.com/file/d/1y0EicuWWsbWIaUm4gU-t_KL96JsZIlqv/view?usp=sharing)

1. **Initializing the Firebase Project and the Firebase Realtime Database:**
   1. Log in to google with a google account and go to <https://firebase.google.com/>
   2. Go to console and add a new project
   3. Under the new project, create a new Realtime database
   4. Click the add button on the root of the Realtime database and add a new value as “employees”.
   5. Under the “employees” root, click add button and write the primary key of an employee record as “e1”. Under that e1, click the add button 4 times and create name, role, email and imageURL key-value pair records for that particular employee record.
   6. Fill out the key-value pairs and these key-value pairs of that particular record will represent a particular employee instance. You can add multiple employee records by clicking the add button under “employees” root and follow the same steps as mentioned above. For the imageURL, you can google any picture and paste the image URL into this field.



1. **Connecting Firebase project with Android Studio project:**
   1. Open Android Studio and create a project with an empty activity. Make sure you use API > 26 as minimum SDK for the project.
   2. Click under Tools in the Android Studio and then Firebase. In the menu, select Realtime Database and select Connect to Database. Once clicked, it will open the firebase console in browser and prompt you to select the firebase project that you want to connect to with your Android Studio project. Select the project we just created in the above steps and select Connect. After connecting, go back to Android Studio and select “Add the Realtime Database SDK to your app” from the opened firebase menu and finally select “Accept Changes”. Hence, your Android Studio project is now connected with your firebase project.
   3. Go to your firebase project in firebase console and navigate to project settings. In the project settings, under the Tab “Your apps”, select google-services.json and download the .json file. Copy the downloaded file and paste it in the android studio project under the app directory.



1. **Importing modules into the Android Studio project:**

We will be using Firebase UI, Glide, CardView, DialogPlus and CircleViewImage modules in our app. So, in order to implement them, we need to put dependencies of each of the modules into our build.gradle file. Add these additional dependencies to the existing dependencies and synchronize your application with the new dependencies. The additional dependencies are:

dependencies **{** implementation 'com.firebaseui:firebase-ui-database:8.0.0'  
 implementation 'de.hdodenhof:circleimageview:3.1.0'  
 implementation 'com.github.bumptech.glide:glide:4.12.0'  
 annotationProcessor 'com.github.bumptech.glide:compiler:4.12.0'  
 implementation 'androidx.recyclerview:recyclerview:1.1.0'  
 implementation "androidx.cardview:cardview:1.0.0"  
 implementation 'com.orhanobut:dialogplus:1.11@aar'}

apply plugin: 'com.google.gms.google-services'  
apply plugin: 'com.android.application'

These third-party dependencies are retrieved from these sites as follows:

<https://firebaseopensource.com/projects/firebase/firebaseui-android/database/readme/>

<https://github.com/hdodenhof/CircleImageView>

<https://github.com/bumptech/glide>

<https://github.com/orhanobut/dialogplus>

Updated dependencies code:

dependencies **{** implementation 'com.firebaseui:firebase-ui-database:8.0.0'  
 implementation 'de.hdodenhof:circleimageview:3.1.0'  
 implementation 'com.github.bumptech.glide:glide:4.12.0'  
 annotationProcessor 'com.github.bumptech.glide:compiler:4.12.0'  
 implementation 'androidx.recyclerview:recyclerview:1.1.0'  
 implementation "androidx.cardview:cardview:1.0.0"  
 implementation 'com.orhanobut:dialogplus:1.11@aar'  
  
  
 implementation 'androidx.appcompat:appcompat:1.4.0'  
 implementation 'com.google.android.material:material:1.4.0'  
 implementation 'androidx.constraintlayout:constraintlayout:2.1.2'  
 implementation 'com.google.firebase:firebase-database:19.2.1'  
 testImplementation 'junit:junit:4.+'  
 androidTestImplementation 'androidx.test.ext:junit:1.1.3'  
 androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'  
**}**apply plugin: 'com.google.gms.google-services'  
apply plugin: 'com.android.application'

1. Go to AndroidManifest.xml and give permission to the user to access internet through the android application.
2. In order to display the records from the database, we can use a recyclerview to display the records, where an item in the recyclerview will represent a record in the employees table in the firebase Realtime database. Add a recyclerview element in the activity\_main.xml and add appropriate constraints to fit the layout in the parent display. Go to MainActivity and register the recyclerview in the OnCreate section of MainActivity.

activity\_main.xml

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.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="#CBD6DA"  
 tools:context=".MainActivity">  
  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/recyclerView"  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
  
 android:layout\_marginStart="1dp"  
 android:layout\_marginEnd="1dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <com.google.android.material.floatingactionbutton.FloatingActionButton  
 android:id="@+id/floatingActionButton"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:backgroundTint="#009688"  
 android:clickable="true"  
 app:layout\_constraintBottom\_toBottomOf="@+id/recyclerView"  
 app:layout\_constraintEnd\_toEndOf="@+id/recyclerView"  
 app:layout\_constraintHorizontal\_bias="0.957"  
 app:layout\_constraintStart\_toStartOf="@+id/recyclerView"  
 app:layout\_constraintTop\_toTopOf="@+id/recyclerView"  
 app:layout\_constraintVertical\_bias="0.976"  
 app:srcCompat="@drawable/add\_icon"  
 tools:ignore="SpeakableTextPresentCheck" />  
</androidx.constraintlayout.widget.ConstraintLayout>

Add an xml file for add icon under res/drawable

add\_icon.xml

<!--add icon for the floating add button-->  
<vector xmlns:android="http://schemas.android.com/apk/res/android"  
 android:width="24dp"  
 android:height="24dp"  
 android:viewportWidth="24"  
 android:viewportHeight="24"  
 android:tint="?attr/colorControlNormal">  
 <path  
 android:fillColor="@android:color/white"  
 android:pathData="M19,13h-6v6h-2v-6H5v-2h6V5h2v6h6v2z"/>  
</vector>

MainActivity.java

package com.example.firebaseproject;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.os.Bundle;  
  
public class MainActivity extends AppCompatActivity {  
 RecyclerView recyclerView;

@Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);

// finding recyclerview  
 recyclerView = (RecyclerView) findViewById(R.id.*recyclerView*);  
  
 // setting layout manager to the recyclerview  
 recyclerView.setLayoutManager(new LinearLayoutManager(this));

}  
}

1. Create a new Java class in the app and name it “EmployeeClass” and inside this class, instantiate the values matching the key names of the Realtime database. Assign setters, getters and constructors to the class.

EmployeeClass.java

package com.example.firebaseproject;  
  
// Employee class with its getters, setters and constructors.  
public class EmployeeClass {  
 String name, role, email, imageURL;  
  
 EmployeeClass(){  
  
 }  
  
 public EmployeeClass(String name, String role, String email, String imageURL) {  
 this.name = name;  
 this.role = role;  
 this.email = email;  
 this.imageURL = imageURL;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public void setName(String name) {  
 this.name = name;  
 }  
  
 public String getRole() {  
 return role;  
 }  
  
 public void setRole(String role) {  
 this.role = role;  
 }  
  
 public String getEmail() {  
 return email;  
 }  
  
 public void setEmail(String email) {  
 this.email = email;  
 }  
  
 public String getImageURL() {  
 return imageURL;  
 }  
  
 public void setImageURL(String imageURL) {  
 this.imageURL = imageURL;  
 }  
}

1. Under the layout directory, create an xml file named “main\_item.xml”. This layout file will represent the layout of a particular item in the recyclerview. We will use a Cardview for the layout of a particular item of the recyclerview. Copy paste the layout file code.

main\_item.xml

<?xml version="1.0" encoding="utf-8"?>  
<androidx.cardview.widget.CardView xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 app:cardCornerRadius="6dp"  
 android:elevation="6dp"  
 app:cardUseCompatPadding="true">  
  
 <RelativeLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:padding="15dp">  
  
 <de.hdodenhof.circleimageview.CircleImageView  
 android:id="@+id/circleImage"  
 android:layout\_width="80dp"  
 android:layout\_height="80dp"  
 android:src="@mipmap/ic\_launcher"  
 app:civ\_border\_color="#FF000000"  
 app:civ\_border\_width="2dp" />  
  
 <TextView  
 android:id="@+id/nameText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="10dp"  
 android:layout\_toRightOf="@+id/circleImage"  
 android:text="@string/employee\_name\_text"  
 android:textColor="@color/black"  
 android:textSize="25sp"  
 android:textStyle="bold" />  
  
 <TextView  
 android:id="@+id/roleText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@+id/nameText"  
 android:layout\_marginLeft="10dp"  
 android:layout\_toRightOf="@+id/circleImage"  
 android:text="@string/employee\_role"  
 android:textColor="@color/black"  
 android:textSize="18sp" />  
  
 <TextView  
 android:id="@+id/emailText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@+id/roleText"  
 android:layout\_marginLeft="10dp"  
 android:layout\_toRightOf="@+id/circleImage"  
 android:text="@string/employee\_email\_address"  
 android:textColor="@color/black"  
 android:textSize="18sp" />  
  
 <LinearLayout  
 android:layout\_below="@+id/emailText"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="5dp"  
 android:orientation="horizontal">  
  
 <Button  
 android:id="@+id/EditButton"  
 android:layout\_width="155dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginRight="5dp"  
 android:text="@string/btn\_edit\_text" />  
  
 <Button  
 android:id="@+id/DeleteButton"  
 android:layout\_width="155dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginRight="5dp"  
 android:text="@string/btn\_delete\_text" />  
 </LinearLayout>  
  
 </RelativeLayout>  
  
  
</androidx.cardview.widget.CardView>

1. We will create a new Java class and name it RecyclerAdapter and we will inherit FirebaseRecyclerAdapter for this class. We will create a recyclerview viewholder class and instantiate all the elements that we are getting from the main\_item.xml layout file in the viewholder class. After creating the view holder class, we will pass the viewholder and the EmployeeClass to the parameter of inherited FirebaseRecyclerAdapter. We will then create the override methods called onBindViewHolder and onCreateViewHolder. Under the onCreateViewHolder, we will register the main\_item.xml as the layout for a particular item of the recyclerview. Under the onBindViewHolder method, we will assign the values of the elements in the itemView of the recyclerview, corresponding to the values of the Employee Class, imported from the Realtime database. We are using Glide module to display image into our application through an image URL. Hence, our Adapter class for the recyclerview is now ready.

RecyclerAdapter.java

package com.example.firebaseproject;  
  
import android.app.AlertDialog;  
import android.content.DialogInterface;  
import android.text.Layout;  
import android.view.Gravity;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import androidx.annotation.NonNull;  
import androidx.recyclerview.widget.RecyclerView;  
  
import com.bumptech.glide.Glide;  
import com.firebase.ui.database.FirebaseRecyclerAdapter;  
import com.firebase.ui.database.FirebaseRecyclerOptions;  
import com.google.android.gms.tasks.OnFailureListener;  
import com.google.android.gms.tasks.OnSuccessListener;  
import com.google.firebase.database.FirebaseDatabase;  
import com.orhanobut.dialogplus.DialogPlus;  
import com.orhanobut.dialogplus.ViewHolder;  
  
import java.util.HashMap;  
import java.util.Map;  
  
import de.hdodenhof.circleimageview.CircleImageView;  
  
public class RecyclerAdapter extends FirebaseRecyclerAdapter<EmployeeClass, RecyclerAdapter.ViewHolder> {  
  
 // initializing a recycler view adapter that inherits FirebaseRecyclerAdapter  
 */\*\*  
 \* Initialize a {****@link*** *RecyclerView.Adapter} that listens to a Firebase query. See  
 \* {****@link*** *FirebaseRecyclerOptions} for configuration options.  
 \*  
 \** ***@param*** *options  
 \*/* // constructor for the recyclerview adapter  
 public RecyclerAdapter(@NonNull FirebaseRecyclerOptions<EmployeeClass> options) {  
 super(options);  
 }  
  
 @Override  
 protected void onBindViewHolder(@NonNull ViewHolder holder, final int position, @NonNull EmployeeClass model) {  
 // setting employee itemView with the imported employee instance from firebase realtime database  
 holder.name.setText(model.getName());  
 holder.role.setText(model.getRole());  
 holder.email.setText(model.getEmail());  
  
 // using a third party Glide module to display an image into an android application from a URL  
 Glide.*with*(holder.img.getContext())  
 .load(model.getImageURL())  
 .placeholder(R.drawable.*common\_google\_signin\_btn\_icon\_dark*)  
 .circleCrop()  
 .error(R.drawable.*common\_google\_signin\_btn\_icon\_dark\_normal*)  
 .into(holder.img);  
  
  
 }  
  
 // assigning the main\_item.xml as the layout of every item in the recyclerview  
 @NonNull  
 @Override  
 public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
 View view = LayoutInflater.*from*(parent.getContext()).inflate(R.layout.*main\_item*,parent,false);  
  
 return new ViewHolder(view);  
 }  
  
  
 // view holder that contains all the elements of an item in a recyclerview and initialized  
 class ViewHolder extends RecyclerView.ViewHolder{  
  
 // initializing  
 CircleImageView img;  
 TextView name, role, email;  
   
  
 public ViewHolder(@NonNull View itemView) {  
 super(itemView);  
  
 // finding the elements from xml file and assigning them to variables  
 img = (CircleImageView) itemView.findViewById(R.id.*circleImage*);  
 name = (TextView) itemView.findViewById(R.id.*nameText*);  
 role = (TextView) itemView.findViewById(R.id.*roleText*);  
 email = (TextView) itemView.findViewById(R.id.*emailText*);  
  
 }  
 }  
}

1. Go to <https://firebaseopensource.com/projects/firebase/firebaseui-android/database/readme/>

and copy a code snippet:

FirebaseRecyclerOptions<Chat> options =

new FirebaseRecyclerOptions.Builder<Chat>()

.setQuery(query, Chat.class)

.build();

Modify this code snippet to fit with our EmployeeClass and paste it on the OnCreate portion of MainActivity.java:

FirebaseRecyclerOptions<EmployeeClass> options =  
 new FirebaseRecyclerOptions.Builder<EmployeeClass>()  
 .setQuery(FirebaseDatabase.*getInstance*().getReference().child("employees"), EmployeeClass.class)  
 .build();

1. We will instantiate the adapter and set the adapter to our recyclerview. We will create two override methods, i.e. onStart() and onStop(), these methods will read records from the Realtime database in Realtime and update the recyclerview if there’s any changes.

MainActivity.java (Updated version)

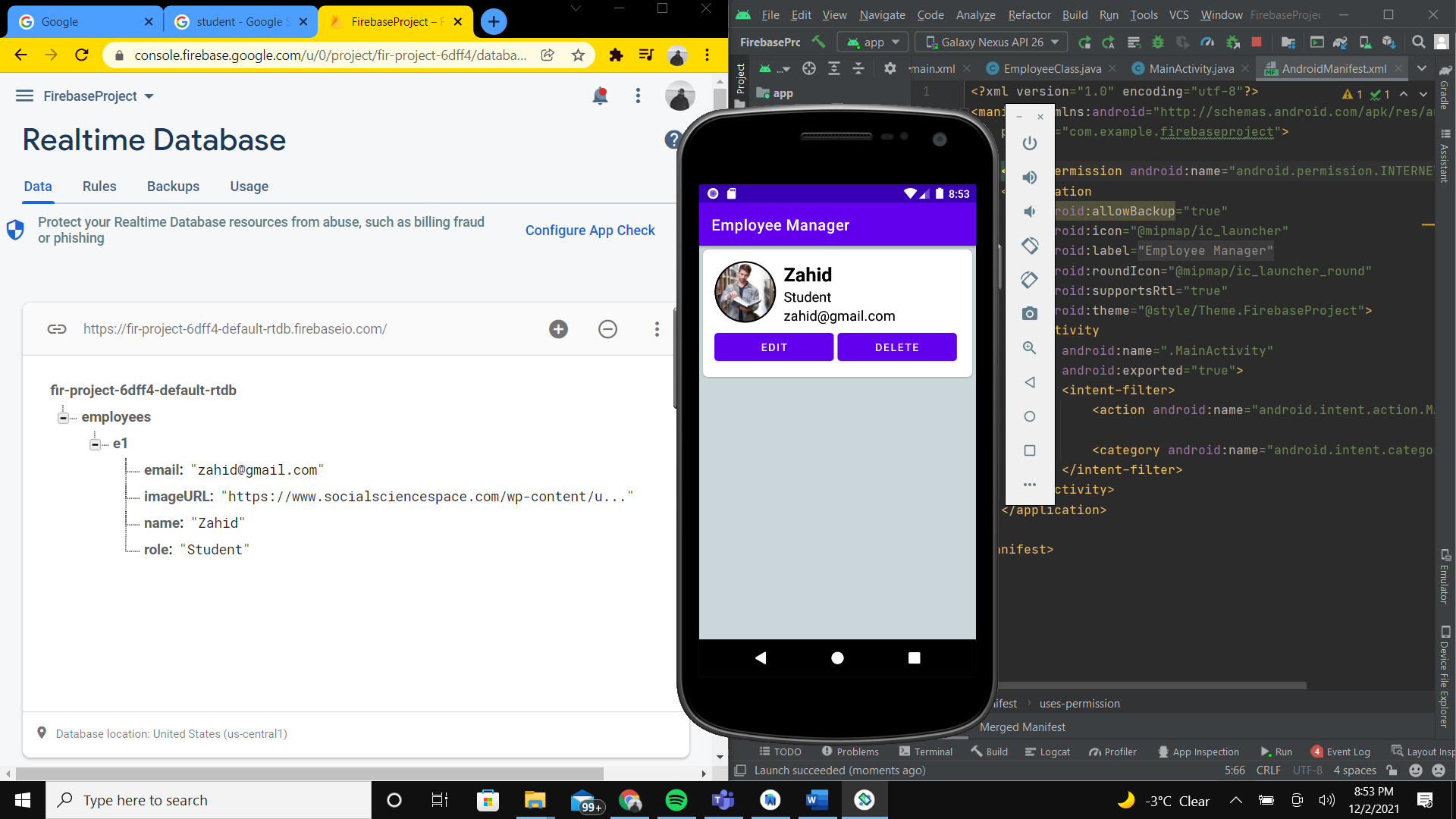
package com.example.firebaseproject;  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
  
import com.firebase.ui.database.FirebaseRecyclerOptions;  
import com.google.android.material.floatingactionbutton.FloatingActionButton;  
import com.google.firebase.database.FirebaseDatabase;  
  
public class MainActivity extends AppCompatActivity {  
  
 // initializing elements  
 RecyclerView recyclerView;  
 RecyclerAdapter recyclerAdapter;  
 FloatingActionButton floatingActionButton;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 // finding recyclerview  
 recyclerView = (RecyclerView) findViewById(R.id.*recyclerView*);  
  
 // setting layout manager to the recyclerview  
 recyclerView.setLayoutManager(new LinearLayoutManager(this));  
  
 // calling out firebase realtime database to retrieve data  
 // setting the employee class with the imported employee instance from the database  
 // all the retrieved employee instances are inside the FirebaseRecyclerOptions variable 'options'  
 FirebaseRecyclerOptions<EmployeeClass> options =  
 new FirebaseRecyclerOptions.Builder<EmployeeClass>()  
 .setQuery(FirebaseDatabase.*getInstance*().getReference().child("employees"), EmployeeClass.class)  
 .build();  
  
 // setting the custom recyclerview adapter to the recyclerview with the imported 'options' variable  
 recyclerAdapter = new RecyclerAdapter(options);  
 recyclerView.setAdapter(recyclerAdapter);  
  
  
  
 }  
  
  
 // application is listening in real time for any changes in real time database and updating the recyclerview adapter  
 @Override  
 protected void onStart() {  
 super.onStart();  
 recyclerAdapter.startListening();  
 }  
  
 // application is now stopped listening in real time for any changes in real time database and updating the recyclerview adapter  
 @Override  
 protected void onStop() {  
 super.onStop();  
 recyclerAdapter.stopListening();  
 }  
}

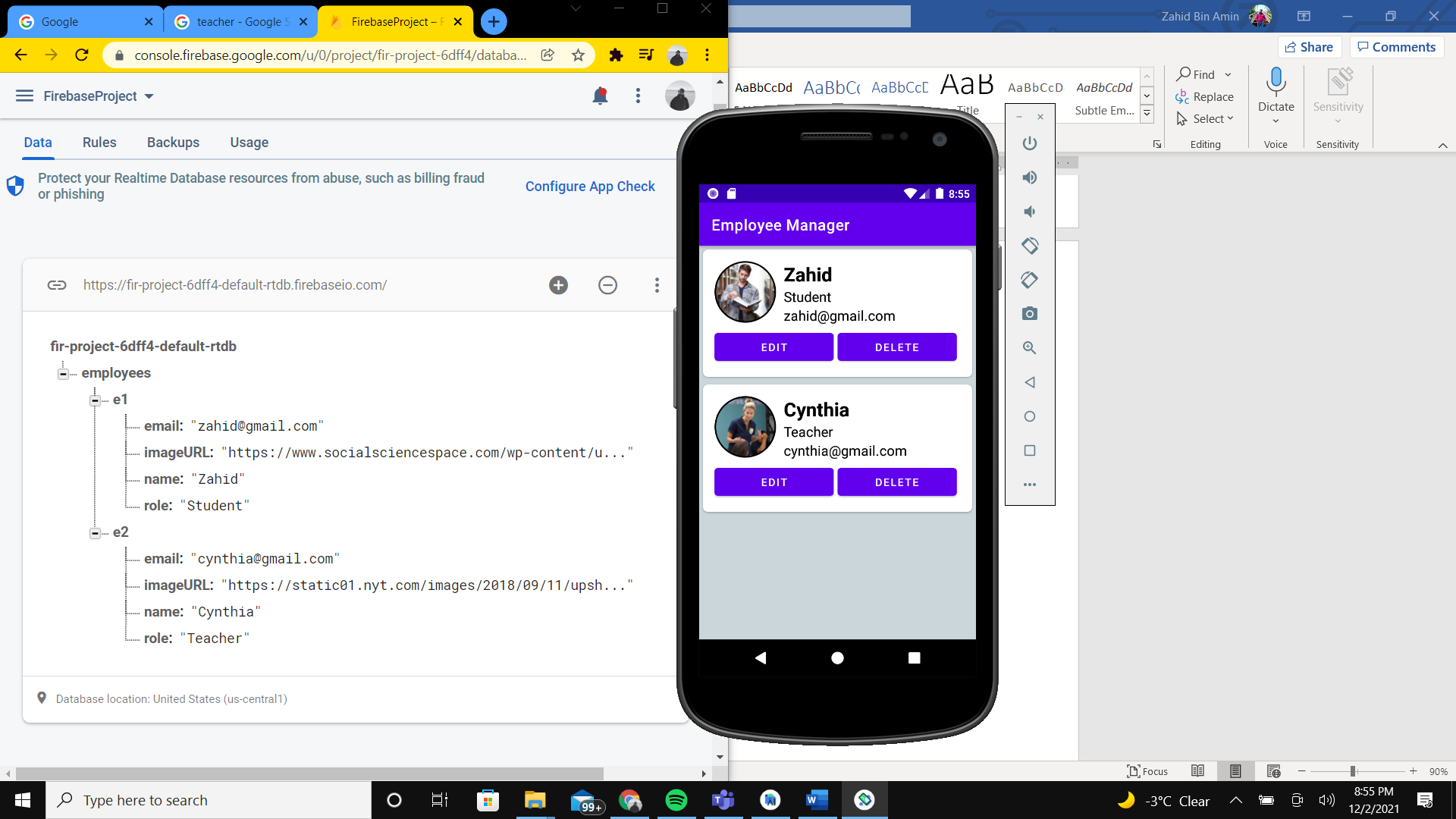
11. Update the strings.xml under res directory since we don’t want to hardcode strings into our layout files

strings.xml

<resources>  
 <string name="app\_name">Employee Manager</string>  
 <string name="add\_new\_employee">Add New Employee</string>  
 <string name="button\_add\_text">Add</string>  
 <string name="employee\_name\_text">Employee Name</string>  
 <string name="employee\_role">Role</string>  
 <string name="employee\_email\_address">Email Address</string>  
 <string name="btn\_edit\_text">Edit</string>  
 <string name="btn\_delete\_text">Delete</string>  
 <string name="update\_popup\_title">Update Details</string>  
 <string name="employee\_name">Name</string>  
 <string name="employee\_profile\_url">Profile URL</string>  
 <string name="btn\_update\_txt">Update</string>  
</resources>

After coding the activity\_main.xml, main\_item.xml, MainActivity.java, RecyclerAdapter.java and EmployeeClass.java, build and run the app. You should be able to see all the records in application fetched from the Realtime database. If the app does not run, try wiping data from avd manager and re run the application again. Sometimes the emulator is clogged up with previous data and as a result, it has a hard time running a new application.





Let’s add the functionality of the add floating button. Copy this code and paste it on the OnCreate method of MainActivity.java. We are listening for click events in this add button and every time a click event happens, AddActivity.java is started. After the code for MainActivity.java, is the code for AddActivity.java. Create a class named AddActivity and copy paste the code in that java class.

// finding the add floating button and assigning an OnClick listener  
floatingActionButton = (FloatingActionButton) findViewById(R.id.*floatingActionButton*);  
  
// every time we click on this add button, it takes us to the AddActivity activity from MainActivity  
floatingActionButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent intent = new Intent(MainActivity.this, AddActivity.class);  
 startActivity(intent);  
  
 }  
});

MainActivity.java (Updated)

package com.example.firebaseproject;  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
  
import com.firebase.ui.database.FirebaseRecyclerOptions;  
import com.google.android.material.floatingactionbutton.FloatingActionButton;  
import com.google.firebase.database.FirebaseDatabase;  
  
public class MainActivity extends AppCompatActivity {  
  
 // initializing elements  
 RecyclerView recyclerView;  
 RecyclerAdapter recyclerAdapter;  
 FloatingActionButton floatingActionButton;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 // finding recyclerview  
 recyclerView = (RecyclerView) findViewById(R.id.*recyclerView*);  
  
 // setting layout manager to the recyclerview  
 recyclerView.setLayoutManager(new LinearLayoutManager(this));  
  
 // calling out firebase realtime database to retrieve data  
 // setting the employee class with the imported employee instance from the database  
 // all the retrieved employee instances are inside the FirebaseRecyclerOptions variable 'options'  
 FirebaseRecyclerOptions<EmployeeClass> options =  
 new FirebaseRecyclerOptions.Builder<EmployeeClass>()  
 .setQuery(FirebaseDatabase.*getInstance*().getReference().child("employees"), EmployeeClass.class)  
 .build();  
  
 // setting the custom recyclerview adapter to the recyclerview with the imported 'options' variable  
 recyclerAdapter = new RecyclerAdapter(options);  
 recyclerView.setAdapter(recyclerAdapter);  
  
 // finding the add floating button and assigning an OnClick listener  
 floatingActionButton = (FloatingActionButton) findViewById(R.id.*floatingActionButton*);  
  
 // every time we click on this add button, it takes us to the AddActivity activity from MainActivity  
 floatingActionButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent intent = new Intent(MainActivity.this, AddActivity.class);  
 startActivity(intent);  
  
 }  
 });  
  
 }  
  
  
 // application is listening in real time for any changes in real time database and updating the recyclerview adapter  
 @Override  
 protected void onStart() {  
 super.onStart();  
 recyclerAdapter.startListening();  
 }  
  
 // application is now stopped listening in real time for any changes in real time database and updating the recyclerview adapter  
 @Override  
 protected void onStop() {  
 super.onStop();  
 recyclerAdapter.stopListening();  
 }  
}

AddActivity.java

package com.example.firebaseproject;  
  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.Toast;  
  
import com.google.android.gms.tasks.OnFailureListener;  
import com.google.android.gms.tasks.OnSuccessListener;  
import com.google.firebase.database.FirebaseDatabase;  
  
import java.util.HashMap;  
import java.util.Map;  
  
public class AddActivity extends AppCompatActivity {  
  
 // initializing  
 EditText name, role,email,imageURL;  
 Button save;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_add);  
  
 // finding all the elements and assigning them to variables  
 name = (EditText) findViewById(R.id.NameInput);  
 role = (EditText) findViewById(R.id.RoleInput);  
 email = (EditText) findViewById(R.id.EmailInput);  
 imageURL = (EditText) findViewById(R.id.ImageInput);  
  
 save = (Button) findViewById(R.id.AddButton);  
  
 // listening for click events for the save button and performing AddData() function for every click event  
 save.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 AddData();  
 }  
 });  
 }  
  
 private void AddData(){  
 // input validation check  
 if (name.getText().toString().equals("") || role.getText().toString().equals("") || email.getText().toString().equals("")){  
 Toast.*makeText*(AddActivity.this, "Please fill out name, role and email fields before adding", Toast.*LENGTH\_SHORT*).show();  
 }  
 // if input validation passes then data from the form will be used to add a new employee instance  
 else {  
 // creating a hashmap and setting 4 different key value pairs  
 Map<String, Object> map = new HashMap<>();  
 map.put("name", name.getText().toString());  
 map.put("role", role.getText().toString());  
 map.put("email", email.getText().toString());  
 map.put("imageURL", imageURL.getText().toString());  
  
 // calling the firebase realtime database and pushing the new hashmap into the employee root  
 FirebaseDatabase.*getInstance*().getReference().child("employees").push()  
 .setValue(map)  
 .addOnSuccessListener(new OnSuccessListener<Void>() {  
  
 // if add was successful, it will show a toast message that is successful and clear out the form and redirect to the  
 // Main Activity  
 @Override  
 public void onSuccess(Void unused) {  
 Toast.*makeText*(AddActivity.this, "Added successfully", Toast.*LENGTH\_SHORT*).show();  
  
 name.setText("");  
 role.setText("");  
 email.setText("");  
 imageURL.setText("");  
 startActivity(new Intent(getApplicationContext(), MainActivity.class));  
 }  
 })  
 .addOnFailureListener(new OnFailureListener() {  
  
 // if add was not successful, it will show a toast message that add failed  
 @Override  
 public void onFailure(@NonNull Exception e) {  
 Toast.*makeText*(AddActivity.this, "Error in adding new employee record", Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
 }  
 }  
  
 // if we pressed the back button, it will take us back to the main activity  
 @Override  
 public void onBackPressed() {  
 startActivity(new Intent(getApplicationContext(), MainActivity.class));  
 // do something on back.  
  
 }  
}

Then we have to write code for the layout file for the AddActivity.java.

activity\_add.xml

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 >  
  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:background="@color/purple\_500"  
 android:gravity="center"  
 android:text="@string/add\_new\_employee"  
 android:textColor="#5D4037"  
 android:textSize="25sp" />  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:gravity="center"  
 android:textSize="20sp"  
 android:layout\_marginTop="10dp"  
 android:layout\_marginBottom="5dp"  
 android:text="@string/employee\_name"/>  
  
 <EditText  
 android:id="@+id/NameInput"  
 android:layout\_width="match\_parent"  
 android:layout\_height="48dp"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginRight="10dp"  
 android:gravity="center"  
 tools:ignore="SpeakableTextPresentCheck" />  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:gravity="center"  
 android:textSize="20sp"  
 android:layout\_marginTop="10dp"  
 android:layout\_marginBottom="5dp"  
 android:text="@string/employee\_role"/>  
  
 <EditText  
 android:id="@+id/RoleInput"  
 android:layout\_width="match\_parent"  
 android:layout\_height="48dp"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginRight="10dp"  
 android:gravity="center"  
 tools:ignore="SpeakableTextPresentCheck" />  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:gravity="center"  
 android:textSize="20sp"  
 android:layout\_marginTop="10dp"  
 android:layout\_marginBottom="5dp"  
 android:text="@string/employee\_email\_address"/>  
  
 <EditText  
 android:id="@+id/EmailInput"  
 android:layout\_width="match\_parent"  
 android:layout\_height="48dp"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginRight="10dp"  
 android:gravity="center"  
 tools:ignore="SpeakableTextPresentCheck" />  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:gravity="center"  
 android:textSize="20sp"  
 android:layout\_marginTop="10dp"  
 android:layout\_marginBottom="5dp"  
 android:text="@string/employee\_profile\_url"/>  
  
 <EditText  
 android:id="@+id/ImageInput"  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginRight="10dp"  
 android:gravity="center"  
 tools:ignore="SpeakableTextPresentCheck" />  
 <Button  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="@string/button\_add\_text"  
 android:layout\_marginLeft="25dp"  
 android:layout\_marginRight="25dp"  
 android:layout\_marginTop="10dp"  
 android:textSize="21sp"  
 android:gravity="center"  
 android:id="@+id/AddButton"/>  
  
</LinearLayout>

Now try running the app and the application will allow you to successfully add data into the realtime database.

Now let’s implement the functionalities of the edit and delete buttons of an item in the recyclerview.

Insert this code snippet in the OnBindViewHolder method of RecyclerAdapter.java and initialize the buttons on viewholder of the RecyclerAdapter class.

// edit button is listening for click events  
holder.editButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
  
 // using a third party DialogPlus module to generate a custom made dialog popup generated from an xml file  
 // this dialog popup will show up when edit button is clicked  
 final DialogPlus dialogPlus = DialogPlus.*newDialog*(holder.img.getContext())  
 .setContentHolder(new com.orhanobut.dialogplus.ViewHolder(R.layout.*update\_popup*))  
 .setGravity(Gravity.*BOTTOM*)  
 .create();  
  
 // finding elements from xml file that the DialogPlus uses to generate the dialog popup  
 View view = dialogPlus.getHolderView();  
 EditText name = view.findViewById(R.id.*NameInput*);  
 EditText role = view.findViewById(R.id.*RoleInput*);  
 EditText email = view.findViewById(R.id.*EmailInput*);  
 EditText imageURL = view.findViewById(R.id.*ImageInput*);  
 Button updateButton = view.findViewById(R.id.*updateButton*);  
  
 // assigning the values from the imported employee instance from the database to the elements  
 name.setText(model.getName());  
 role.setText(model.getRole());  
 email.setText(model.getEmail());  
 imageURL.setText(model.getImageURL());  
  
 // dialogPlus is displayed  
 dialogPlus.show();  
  
 // update button is listening for click events  
 updateButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 // form input validation  
 if (name.getText().toString().equals("") || role.getText().toString().equals("") || email.getText().toString().equals("")){  
 Toast.*makeText*(holder.name.getContext(), "Please fill out name, role and email fields before updating", Toast.*LENGTH\_SHORT*).show();  
 }  
  
 // if validation passes  
 else {  
 // creating a hashmap and setting 4 different key value pairs  
 Map<String, Object> map = new HashMap<>();  
 map.put("name", name.getText().toString());  
 map.put("role", role.getText().toString());  
 map.put("email", email.getText().toString());  
 map.put("imageURL", imageURL.getText().toString());  
  
 // calling the firebase realtime database and look for an employee record  
 // matching the key of the employee item of the recyclerview  
 // and then passing the updated hashmap to replace that particular employee record in database  
 FirebaseDatabase.*getInstance*().getReference().child("employees")  
 .child(getRef(position).getKey()).updateChildren(map)  
 .addOnSuccessListener(new OnSuccessListener<Void>() {  
  
 // if update operation succeeds, it will show a friendly toast message to the user  
 // the dialog popup will go away  
 @Override  
 public void onSuccess(Void unused) {  
 Toast.*makeText*(holder.name.getContext(), "Data updated successfully.", Toast.*LENGTH\_SHORT*).show();  
 dialogPlus.dismiss();  
 }  
 })  
 .addOnFailureListener(new OnFailureListener() {  
  
 // if update operation fail, it will show a friendly toast message that operation has failed  
 @Override  
 public void onFailure(@NonNull Exception e) {  
 Toast.*makeText*(holder.name.getContext(), "Error while updating", Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
 }  
 }  
 });  
  
 }  
});

// delete button is listening for click events  
holder.deleteButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
  
 // when button clicked, it will show an alert dialog taking confirmation from the user  
 AlertDialog.Builder builder = new AlertDialog.Builder(holder.name.getContext());  
 builder.setTitle("Are you sure you want to delete?");  
 builder.setMessage("Deleted data will can not be restored");  
  
 // if user confirms the delete  
 builder.setPositiveButton("Delete", new DialogInterface.OnClickListener() {  
 @Override  
 public void onClick(DialogInterface dialog, int which) {  
  
 // calling the firebase realtime database to look for a key (of the selected itemView of recyclerview)  
 // in the 'employee' root and remove the value  
 // employee record is deleted permanently from both the database and the application  
 FirebaseDatabase.*getInstance*().getReference().child("employees").child(getRef(position).getKey()).removeValue();  
 }  
 });  
  
 // if user changes their mind  
 builder.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {  
 @Override  
 public void onClick(DialogInterface dialog, int which) {  
  
 // shows a friendly message that delete operation is cancelled  
 Toast.*makeText*(holder.name.getContext(), "Cancelled", Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
 builder.show();  
 }  
});

RecyclerAdapter.java (Updated)

package com.example.firebaseproject;  
  
import android.app.AlertDialog;  
import android.content.DialogInterface;  
import android.text.Layout;  
import android.view.Gravity;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import androidx.annotation.NonNull;  
import androidx.recyclerview.widget.RecyclerView;  
  
import com.bumptech.glide.Glide;  
import com.firebase.ui.database.FirebaseRecyclerAdapter;  
import com.firebase.ui.database.FirebaseRecyclerOptions;  
import com.google.android.gms.tasks.OnFailureListener;  
import com.google.android.gms.tasks.OnSuccessListener;  
import com.google.firebase.database.FirebaseDatabase;  
import com.orhanobut.dialogplus.DialogPlus;  
import com.orhanobut.dialogplus.ViewHolder;  
  
import java.util.HashMap;  
import java.util.Map;  
  
import de.hdodenhof.circleimageview.CircleImageView;  
  
public class RecyclerAdapter extends FirebaseRecyclerAdapter<EmployeeClass, RecyclerAdapter.ViewHolder> {  
  
 // initializing a recycler view adapter that inherits FirebaseRecyclerAdapter  
 */\*\*  
 \* Initialize a {****@link*** *RecyclerView.Adapter} that listens to a Firebase query. See  
 \* {****@link*** *FirebaseRecyclerOptions} for configuration options.  
 \*  
 \** ***@param*** *options  
 \*/* // constructor for the recyclerview adapter  
 public RecyclerAdapter(@NonNull FirebaseRecyclerOptions<EmployeeClass> options) {  
 super(options);  
 }  
  
 @Override  
 protected void onBindViewHolder(@NonNull ViewHolder holder, final int position, @NonNull EmployeeClass model) {  
 // setting employee itemView with the imported employee instance from firebase realtime database  
 holder.name.setText(model.getName());  
 holder.role.setText(model.getRole());  
 holder.email.setText(model.getEmail());  
  
 // using a third party Glide module to display an image into an android application from a URL  
 Glide.*with*(holder.img.getContext())  
 .load(model.getImageURL())  
 .placeholder(R.drawable.*common\_google\_signin\_btn\_icon\_dark*)  
 .circleCrop()  
 .error(R.drawable.*common\_google\_signin\_btn\_icon\_dark\_normal*)  
 .into(holder.img);  
  
 // delete button is listening for click events  
 holder.deleteButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
  
 // when button clicked, it will show an alert dialog taking confirmation from the user  
 AlertDialog.Builder builder = new AlertDialog.Builder(holder.name.getContext());  
 builder.setTitle("Are you sure you want to delete?");  
 builder.setMessage("Deleted data will can not be restored");  
  
 // if user confirms the delete  
 builder.setPositiveButton("Delete", new DialogInterface.OnClickListener() {  
 @Override  
 public void onClick(DialogInterface dialog, int which) {  
  
 // calling the firebase realtime database to look for a key (of the selected itemView of recyclerview)  
 // in the 'employee' root and remove the value  
 // employee record is deleted permanently from both the database and the application  
 FirebaseDatabase.*getInstance*().getReference().child("employees").child(getRef(position).getKey()).removeValue();  
 }  
 });  
  
 // if user changes their mind  
 builder.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {  
 @Override  
 public void onClick(DialogInterface dialog, int which) {  
  
 // shows a friendly message that delete operation is cancelled  
 Toast.*makeText*(holder.name.getContext(), "Cancelled", Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
 builder.show();  
 }  
 });  
  
 // edit button is listening for click events  
 holder.editButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
  
 // using a third party DialogPlus module to generate a custom made dialog popup generated from an xml file  
 // this dialog popup will show up when edit button is clicked  
 final DialogPlus dialogPlus = DialogPlus.*newDialog*(holder.img.getContext())  
 .setContentHolder(new com.orhanobut.dialogplus.ViewHolder(R.layout.*update\_popup*))  
 .setGravity(Gravity.*BOTTOM*)  
 .create();  
  
 // finding elements from xml file that the DialogPlus uses to generate the dialog popup  
 View view = dialogPlus.getHolderView();  
 EditText name = view.findViewById(R.id.*NameInput*);  
 EditText role = view.findViewById(R.id.*RoleInput*);  
 EditText email = view.findViewById(R.id.*EmailInput*);  
 EditText imageURL = view.findViewById(R.id.*ImageInput*);  
 Button updateButton = view.findViewById(R.id.*updateButton*);  
  
 // assigning the values from the imported employee instance from the database to the elements  
 name.setText(model.getName());  
 role.setText(model.getRole());  
 email.setText(model.getEmail());  
 imageURL.setText(model.getImageURL());  
  
 // dialogPlus is displayed  
 dialogPlus.show();  
  
 // update button is listening for click events  
 updateButton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 // form input validation  
 if (name.getText().toString().equals("") || role.getText().toString().equals("") || email.getText().toString().equals("")){  
 Toast.*makeText*(holder.name.getContext(), "Please fill out name, role and email fields before updating", Toast.*LENGTH\_SHORT*).show();  
 }  
  
 // if validation passes  
 else {  
 // creating a hashmap and setting 4 different key value pairs  
 Map<String, Object> map = new HashMap<>();  
 map.put("name", name.getText().toString());  
 map.put("role", role.getText().toString());  
 map.put("email", email.getText().toString());  
 map.put("imageURL", imageURL.getText().toString());  
  
 // calling the firebase realtime database and look for an employee record  
 // matching the key of the employee item of the recyclerview  
 // and then passing the updated hashmap to replace that particular employee record in database  
 FirebaseDatabase.*getInstance*().getReference().child("employees")  
 .child(getRef(position).getKey()).updateChildren(map)  
 .addOnSuccessListener(new OnSuccessListener<Void>() {  
  
 // if update operation succeeds, it will show a friendly toast message to the user  
 // the dialog popup will go away  
 @Override  
 public void onSuccess(Void unused) {  
 Toast.*makeText*(holder.name.getContext(), "Data updated successfully.", Toast.*LENGTH\_SHORT*).show();  
 dialogPlus.dismiss();  
 }  
 })  
 .addOnFailureListener(new OnFailureListener() {  
  
 // if update operation fail, it will show a friendly toast message that operation has failed  
 @Override  
 public void onFailure(@NonNull Exception e) {  
 Toast.*makeText*(holder.name.getContext(), "Error while updating", Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
 }  
 }  
 });  
  
 }  
 });  
 }  
  
 // assigning the main\_item.xml as the layout of every item in the recyclerview  
 @NonNull  
 @Override  
 public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
 View view = LayoutInflater.*from*(parent.getContext()).inflate(R.layout.*main\_item*,parent,false);  
  
 return new ViewHolder(view);  
 }  
  
  
 // view holder that contains all the elements of an item in a recyclerview and initialized  
 class ViewHolder extends RecyclerView.ViewHolder{  
  
 // initializing  
 CircleImageView img;  
 TextView name, role, email;  
 Button editButton, deleteButton;  
  
 public ViewHolder(@NonNull View itemView) {  
 super(itemView);  
  
 // finding the elements from xml file and assigning them to variables  
 img = (CircleImageView) itemView.findViewById(R.id.*circleImage*);  
 name = (TextView) itemView.findViewById(R.id.*nameText*);  
 role = (TextView) itemView.findViewById(R.id.*roleText*);  
 email = (TextView) itemView.findViewById(R.id.*emailText*);  
 editButton = (Button) itemView.findViewById(R.id.*EditButton*);  
 deleteButton = (Button) itemView.findViewById(R.id.*DeleteButton*);  
 }  
 }  
}

When the edit button is pressed, a custom made dialog popup is shown on the screen from where the user can edit the employee record. In order to create the custom dialog popup, we have to create a new layout resource file under res/layout.

update\_popup.xml

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical">  
  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:background="@color/purple\_500"  
 android:gravity="center"  
 android:text="@string/update\_popup\_title"  
 android:textColor="#5D4037"  
 android:textSize="25dp" />  
  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="5dp"  
 android:layout\_marginBottom="5dp"  
 android:gravity="center"  
 android:text="@string/employee\_name"  
 android:textSize="20sp" />  
  
 <EditText  
 android:id="@+id/NameInput"  
 android:layout\_width="match\_parent"  
 android:layout\_height="48dp"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginRight="10dp"  
 android:gravity="center"  
 tools:ignore="SpeakableTextPresentCheck" />  
  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="5dp"  
 android:layout\_marginBottom="5dp"  
 android:gravity="center"  
 android:text="@string/employee\_role"  
 android:textSize="20sp" />  
  
 <EditText  
 android:id="@+id/RoleInput"  
 android:layout\_width="match\_parent"  
 android:layout\_height="48dp"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginRight="10dp"  
 android:gravity="center" />  
  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="5dp"  
 android:layout\_marginBottom="5dp"  
 android:gravity="center"  
 android:text="@string/employee\_email\_address"  
 android:textSize="20sp" />  
  
 <EditText  
 android:id="@+id/EmailInput"  
 android:layout\_width="match\_parent"  
 android:layout\_height="48dp"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginRight="10dp"  
 android:gravity="center" />  
  
 <TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="5dp"  
 android:layout\_marginBottom="5dp"  
 android:gravity="center"  
 android:text="@string/employee\_profile\_url"  
 android:textSize="20sp" />  
  
 <EditText  
 android:id="@+id/ImageInput"  
 android:layout\_width="match\_parent"  
 android:layout\_height="50dp"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginRight="10dp"  
 android:gravity="center"  
 tools:ignore="SpeakableTextPresentCheck" />  
  
 <Button  
 android:id="@+id/updateButton"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="25dp"  
 android:layout\_marginTop="10dp"  
 android:layout\_marginRight="25dp"  
 android:layout\_marginBottom="20dp"  
 android:gravity="center"  
 android:text="@string/btn\_update\_txt"  
 android:textSize="21sp" />  
</LinearLayout>

Try running the app now, edit and delete buttons should be functional.

Therefore, the application is now complete.

Thank you for going through my tutorial.<