**push Notification.**

**Since FCM is a Firebase service, you’ll need to add Firebase to the app:**

STEP 1: **Adding firebase to the application.**

1. Head over to the Firebase Console. (Done “Accident Alert Application”)
2. Select “Add project,” and give your project a name. (Done “Accident Alert Application”)
3. Read the terms and conditions. If you’re happy to proceed, then select “I accept…” followed by “Create project.”
4. Select “Add Firebase to your Android app.”
5. Enter your project’s package name, and then click “Register app.”
6. Select “Download google-services.json.”
7. In Android Studio, drag and drop the google-services.json file into your project’s “app” directory.
8. Open project-level build.gradle file and add the following:

JAVA

classpath 'com.google.gms:google-services:4.0.1'

**Open app-level build.gradle file, and add the Google services plugin, plus the dependencies for Firebase Core and FCM:**

PHP

//Add the Google services plugin//

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

…

dependencies {

implementation fileTree(dir: 'libs', include: ['\*.jar'])

//Add Firebase Core//

implementation 'com.google.firebase:firebase-core:16.0.1'

//Add FCM//

implementation 'com.google.firebase:firebase-messaging:17.3.4'

* When prompted, sync your changes.
* Next, you need to let the Firebase Console know that you’ve successfully added Firebase to your project. Install your app on either a physical Android smartphone or tablet, or an Android Virtual Device (AVD).
* Back in the Firebase Console, select “Run app to verify installation.”
* Once Firebase has detected your app, you’ll see a “Congratulations” message. Select “Continue to the console.”

STEP 2: **Sending push notification with Firebase**

1. create FCM notifications using the Notifications Composer, which is available via the Firebase Console:
2. Make sure your app is installed and running in the background, and that your device has an active Internet connection.
3. In the Firebase Console, select “Cloud Messaging” from the left-hand menu.
4. Select “Send your first message.”
5. Give your message a title and some body text, and then click “Next.”
6. Open the “Select app” dropdown, and choose your application from the list.
7. Once you’ve finished editing this section, click “Next.”
8. Assuming you want to send this message immediately, open the “Send to eligible users” dropdown and select “Now.”
9. In the bottom-right of the screen, click “Publish.”
10. Check all the information in the subsequent popup, and if you’re happy to proceed then select “Publish.”

***After a few moments, all the client devices that you targeted should receive this notification in their system tray.***

***Most of the time, FCM notifications will be delivered immediately, but occasionally it may take a few minutes for a message to arrive, so don’t panic if your notification is delayed***

Hope it works!

**New added code for retrieving the location added to the database.**

**Java code**

1 -A class that will temporarily hold the location reference from the database on firebase.

//class

public static class Post {  
  
  public String Location;  
   
  public Post(String Location) {  
    // ...  
  }  
  
}

2- getting the reference to the firebase database Accident-Alert-Appliaction

// Get a reference to our posts  
final FirebaseDatabase database = FirebaseDatabase.getInstance();  
DatabaseReference ref = database.getReference(“https://accident-alert-application.firebaseio.com/”));

3- Retrieve the last added data to the database location specified as the reference onj point #-2

ref.addChildEventListener(new ChildEventListener() {  
  @Override  
  public void onChildAdded(DataSnapshot dataSnapshot, String prevChildKey) {

//instance of the class for holdingthe location data

    Post newPost = dataSnapshot.getValue(Post.class);

//if we had to print the location data on console  
    System.out.println("Location: " + newPost.Location);  
     
  }

**Adding User Authentication**

1- Authenticate the signIN window to identify or create a new user

* With the project suitably configured to make use of Firebase Authentication and the realtime database, code can now be added to the project to perform the user authentication. Edit the SignInActivity.java file and modify it as follows to initiate the authentication process:

**Java code**

package com.ebookfrenzy.realtimedb;

import android.content.Intent;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import com.firebase.ui.auth.ErrorCodes;

import com.firebase.ui.auth.IdpResponse;

import com.firebase.ui.auth.ResultCodes;

import com.firebase.ui.auth.AuthUI;

import com.google.firebase.auth.FirebaseAuth;

import java.util.ArrayList;

import java.util.List;

public class RealtimeDBActivity extends AppCompatActivity {

FirebaseAuth auth;

private static final int REQUEST\_CODE = 101;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

//replace the underlined code with the name of the layout of the //signInActivity

setContentView(R.layout.*activity\_realtime\_db*);

auth = FirebaseAuth.getInstance();

if (auth.getCurrentUser() != null) {

//replace the underlined code with the name of the signINactivity**.Class**

startActivity(new Intent(this, SignedInActivity.class));

finish();

} else {

authenticateUser();

}

}

.

.

}

* the code added above will call a method named authenticateUser() if the current user is not already signed in. The next step is to add this method:

private void authenticateUser() {

List<AuthUI.IdpConfig> providers = new ArrayList<>();

providers.add(

new AuthUI.IdpConfig.Builder(AuthUI.EMAIL\_PROVIDER).build());

startActivityForResult(

AuthUI.getInstance().createSignInIntentBuilder()

.setAvailableProviders(providers)

.setIsSmartLockEnabled(false)

.build(),

REQUEST\_CODE);

}

* When the authentication activity is completed, the onActivityResult() method will be called. Implement this method now to launch the SignedInActivity/MainMenuActivity (name of your Menu Activity)on a successful authentication:

@Override

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

super.onActivityResult(requestCode, resultCode, data);

IdpResponse response = IdpResponse.fromResultIntent(data);

if (requestCode == REQUEST\_CODE) {

if (resultCode == ResultCodes.OK) {

//replace the underlined code with the name of the mainMenuActivity**.Class**

startActivity(new Intent(this, *SignedInActivity*.class));

return;

}

} else {

if (response == null) {

// User cancelled Sign-in

return;

}

if (response.getErrorCode() == ErrorCodes.NO\_NETWORK) {

// Device has no network connection

return;

}

if (response.getErrorCode() == ErrorCodes.UNKNOWN\_ERROR) {

// Unknown error occurred

return;

}

}

}

**Accessing the Database**

* Code now needs to be added to the MainMenuActivity/ SignedInActivity class to obtain both a database reference and the uid of the current user. Load the MainMenuActivity/ SignedInActivity.java file into the editor and modify it as follows:

package com.ebookfrenzy.realtimedb;

import android.content.Intent;

import android.support.annotation.NonNull;

import android.support.v7.app.AppCompatActivity;

import android.os.Bundle;

import android.view.View;

import android.widget.EditText;

import android.widget.Toast;

import com.firebase.ui.auth.AuthUI;

import com.google.android.gms.tasks.OnCompleteListener;

import com.google.android.gms.tasks.Task;

import com.google.firebase.auth.FirebaseAuth;

import com.google.firebase.auth.FirebaseUser;

import com.google.firebase.database.DataSnapshot;

import com.google.firebase.database.DatabaseError;

import com.google.firebase.database.DatabaseReference;

import com.google.firebase.database.FirebaseDatabase;

import com.google.firebase.database.ValueEventListener;

public class SignedInActivity extends AppCompatActivity {

private static FirebaseUser currentUser;

private static final String TAG = "RealtimeDB";

private FirebaseDatabase database;

private DatabaseReference dbRef;

private EditText userText;

.

.

}

* Next, locate and edit the onCreate() method to identify the current user and to obtain the database reference:

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

//replace the underlined code with the name of the layout

setContentView(R.layout.*activity\_signed\_in*);

userText = (EditText) findViewById(R.id.userText);

currentUser =

FirebaseAuth.getInstance().getCurrentUser();

if (currentUser == null) {

startActivity(new Intent(this, RealtimeDBActivity.class));

finish();

return;

}

database = FirebaseDatabase.getInstance();

dbRef = database.getReference("https://accident-alert-application.firebaseio.com/");

dbRef.addValueEventListener(changeListener);

}

* The above code also adds the value event listener to the database reference so that the app will receive notification when changes occur to the data stored in the database. Remaining within the RealtimeDBActivity.java file, add this listener now:

ValueEventListener changeListener = new ValueEventListener() {

@Override

public void onDataChange(DataSnapshot dataSnapshot) {

String change = dataSnapshot.child(

currentUser.getUid()).child("message")

.getValue(String.class);

userText.setText(change);

}

@Override

public void onCancelled(DatabaseError databaseError) {

notifyUser("Database error: " + databaseError.toException());

}

};

* The onDataChange() listener method will be called when the data stored at /data within the database tree changes and is passed a DataSnapshot instance containing the data. The code within this method extracts the String object from the DataSnapshot instance located at the <user id>/message node. This string is then assigned to the EditText field so that it is visible to the user.
* The onCancelled() listener method notifies the user of any errors that have occurred using a method named notifyUser() which also needs to be added to the SignedInActivity class:

private void notifyUser(String message) {

Toast.makeText(SignedInActivity.this, message,

Toast.LENGTH\_SHORT).show();

}

**New added code for retrieving the location added to the database.**

**Java code**

1 -A class that will temporarily hold the location reference from the database on firebase.

//class

public static class Post {  
  
  public String Location;  
   
  public Post(String Location) {  
    // ...  
  }  
  
}

2- getting the reference to the firebase database Accident-Alert-Appliaction

// Get a reference to our posts  
final FirebaseDatabase database = FirebaseDatabase.getInstance();  
DatabaseReference ref = database.getReference(“https://accident-alert-application.firebaseio.com/”));

3- Retrieve the last added data to the database location specified as the reference onj point #-2

ref.addChildEventListener(new ChildEventListener() {  
  @Override  
  public void onChildAdded(DataSnapshot dataSnapshot, String prevChildKey) {

//instance of the class for holdingthe location data

    Post newPost = dataSnapshot.getValue(Post.class);

//if we had to print the location data on console  
    System.out.println("Location: " + newPost.Location);  
     
  }

**“For today lets stop here… These codes are now messing with my head”**