Mobile_ICP12 Report

ICP 12

. ICP Group 3

. Name : Rishmitha Chennupati

. Email: rchxc@umsystem.edu

My Partner Details

. Partner Name : Ashwini Reddy

. Partner Email : akdrw@mail.umkc.edu

. Partner Repository: https://github.com/UMKC-APL-WebMobileProgramming/ICP12-akonidala19

My Video and source code links:

. ICP12 video： https://umsystem.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=704ea4a6-817c-488a-b4f2-adef00948209

. ICP12 task： https://github.com/UMKC-APL-WebMobileProgramming/ICP12-RishmithaChennupati/tree/main/Mobile_Lesson5

GitHub Repository: https://github.com/UMKC-APL-WebMobileProgramming/ICP12-RishmithaChennupati

Lesson Overview:

In this lesson, we are going to discussSQLite and Firebase databases.

Use Case Description:

1. Employee/Employer with SQLite

2. Signup/Signing with FirebaseProgramming elements:SQLite, Firebase,and CRUD operations in both databases.

To-do Tasks:

Task 1: SQLite

1.Open the use case SQLite provided in the source code and understand how the control flows from source code to UI

2.Add the following functionality to the app:

•deleting employee or employer details

•Updating employee details

Task 2: Firebase

1.Open the use case 'Firebase'provided in source code and understand how the control flows from source code to UI

2.Add the following functionalities to the app:

•Log out

•Delete feature

Screenshots:

Task 1: SQLite:

To display employee and employer information, I created a SQLite application. The program now has the ability to delete employee or employer information as well as update employee information. The following is the source code

```
    SimpleCursorAdapter cursorAdapter = new SimpleCursorAdapter(
            context: this, android.R.layout.simple_spinner_item, cursor, adapterCols, adapterRowViews,  flags: 0
    cursorAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
    binding.employerSpinner.setAdapter(cursorAdapter);

    binding.saveButton.setOnClickListener((view) -> { saveToDB(); });

    binding.updateButton.setOnClickListener((v) -> { updateDB(); });
    binding.deleteButton.setOnClickListener((v) -> { deleteFromDB(); });

    binding.searchButton.setOnClickListener((view) -> { readFromDB(); });
```

```
// update the data base record on clicking the update button
private void updateDB() {

    SQLiteDatabase database = new SampleDBSQLiteHelper( context: this).getReadableDatabase();
    ContentValues values = new ContentValues();
    values.put(SampleDBContract.Employee.COLUMN_FIRSTNAME, binding.firstnameEditText.getText().toString());
    values.put(SampleDBContract.Employee.COLUMN_LASTNAME, binding.lastnameEditText.getText().toString());
    values.put(SampleDBContract.Employee.COLUMN_JOB_DESCRIPTION, binding.jobDescEditText.getText().toString());
    values.put(SampleDBContract.Employee.COLUMN_EMPLOYED_DATE, binding.employedEditText.getText().toString());
    values.put(SampleDBContract.Employee.COLUMN_EMPLOYER_ID,
            ((Cursor) binding.employerSpinner.getSelectedItem()).getInt( i: 0));
    String whereClause = SampleDBContract.Employee.COLUMN_FIRSTNAME + " like ? AND " + SampleDBContract.Employee.COLUMN_LASTNAME+ " like ?";
    String[] whereArgs = {"%" + firstnameCurrent + "%", "%" + lastnameCurrent + "%"};

    database.update(SampleDBContract.Employee.TABLE_NAME,values,whereClause,whereArgs);
    Toast.makeText( context: this,  text: "Database Updated Successfully!" + firstnameCurrent + " " + lastnameCurrent, Toast.LENGTH_LONG).show();
    firstnameCurrent = "";
    lastnameCurrent = "";
    readFromDB();
}
```

```
// delete the record from data base on clicking the delete button
private void deleteFromDB() {

    Toast.makeText( context: this,  text: "Record Deleted Successfully!" , Toast.LENGTH_LONG).show();
    readFromDB();
    try {
        Thread.sleep( millis: 1500);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }

    SQLiteDatabase database = new SampleDBSQLiteHelper( context: this).getReadableDatabase();
    String firstname = binding.firstnameEditText.getText().toString();
    String lastname = binding.lastnameEditText.getText().toString();
    String whereClause = SampleDBContract.Employee.COLUMN_FIRSTNAME + " like ? AND " + SampleDBContract.Employee.COLUMN_LASTNAME + " like ?";
    String[] whereArgs = {"%" + firstname + "%", "%" + lastname + "%"};
    database.delete(SampleDBContract.Employee.TABLE_NAME, whereClause, whereArgs);
    binding.firstnameEditText.setText("");
    binding.lastnameEditText.setText("");
    readFromDB();
}
```

```java
private void readFromDB() {
    String firstname = binding.firstnameEditText.getText().toString();
    String lastname = binding.lastnameEditText.getText().toString();

    SQLiteDatabase database = new SampleDBSQLiteHelper( context: this).getReadableDatabase();

    String[] selectionArgs = {"%" + firstname + "%", "%" + lastname + "%"};

    Cursor cursor = database.rawQuery(SampleDBContract.SELECT_EMPLOYEE_WITH_EMPLOYER, selectionArgs);
    binding.recycleView.setAdapter(new SampleJoinRecyclerViewCursorAdapter( context: this, cursor));
    firstnameCurrent = firstname;
    lastnameCurrent = lastname;
}
```

EmployerActivity.java

```java
    binding.saveButton.setOnClickListener((view) -> { saveToDB(); });
    binding.updateButton.setOnClickListener((v) -> { updateDB(); });
    binding.deleteButton.setOnClickListener((view) -> { deleteFromDB(); });

    binding.searchButton.setOnClickListener((view) -> { readFromDB(); });
}

// update the data base employer record on clicking the update button
private void updateDB() {

    SQLiteDatabase database = new SampleDBSQLiteHelper( context: this).getReadableDatabase();
    ContentValues values = new ContentValues();
    values.put(SampleDBContract.Employer.COLUMN_NAME, binding.nameEditText.getText().toString());
    values.put(SampleDBContract.Employer.COLUMN_DESCRIPTION, binding.descEditText.getText().toString());

    String whereClause = SampleDBContract.Employer.COLUMN_NAME + " like ? AND " + SampleDBContract.Employer.COLUMN_DESCRIPTION+ " like ?";
    String[] whereArgs = {"%" + nameCurrent + "%", "%" + descCurrent + "%"};

    database.update(SampleDBContract.Employer.TABLE_NAME,values,whereClause,whereArgs);
    Toast.makeText( context: this,  text: "Database Updated Successfully!" + nameCurrent, Toast.LENGTH_LONG).show();
    nameCurrent = "";
    descCurrent = "";
    readFromDB();

}
```

EmployerActivity.java

```java
}

// delete the employer record from data base on clicking the delete button
private void deleteFromDB() {

    Toast.makeText( context: this,  text: "Record Deleted Successfully!" , Toast.LENGTH_LONG).show();
    readFromDB();
    try {
        Thread.sleep( millis: 1500);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }

    SQLiteDatabase database = new SampleDBSQLiteHelper( context: this).getReadableDatabase();
    String name = binding.nameEditText.getText().toString();
    String desc = binding.descEditText.getText().toString();
    String whereClause = SampleDBContract.Employer.COLUMN_NAME + " like ? AND " + SampleDBContract.Employer.COLUMN_DESCRIPTION + " like ?";
    String[] whereArgs = {"%" + name + "%", "%" + desc + "%"};
    database.delete(SampleDBContract.Employer.TABLE_NAME, whereClause, whereArgs);
    binding.nameEditText.setText("");
    binding.descEditText.setText("");
    readFromDB();

}
```

EmployerActivity.java

```
            SampleDBContract.Employer.COLUMN_NAME + " like ? and " +
                    SampleDBContract.Employer.COLUMN_FOUNDED_DATE + " > ? and " +
                    SampleDBContract.Employer.COLUMN_DESCRIPTION + " like ?";

    String[] selectionArgs = {"%" + name + "%", date + "", "%" + desc + "%"};

    Cursor cursor = database.query(
            SampleDBContract.Employer.TABLE_NAME,     // The table to query
            projection,                               // The columns to return
            selection,                                // The columns for the WHERE clause
            selectionArgs,                            // The values for the WHERE clause
            groupBy: null,                            // don't group the rows
            having: null,                             // don't filter by row groups
            orderBy: null                             // don't sort
    );

    binding.recycleView.setAdapter(new SampleRecyclerViewCursorAdapter( context: this, cursor));
    nameCurrent = name;
    descCurrent = desc;
}
```

In SQLite layout I have written code for update and delete buttons in activity employee and employer xml files.

activity_employee.xml

```
<Button
    android:id="@+id/updateButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@id/employerSpinner"
    android:layout_toLeftOf="@id/saveButton"
    android:layout_marginBottom="30dp"
    android:text="Update" />

<Button
    android:id="@+id/deleteButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@id/employerSpinner"
    android:layout_marginTop="0dp"
    android:layout_marginRight="11dp"
    android:layout_marginBottom="30dp"
    android:layout_toLeftOf="@id/updateButton"
    android:text="Delete" />
```

activity_employer.xml

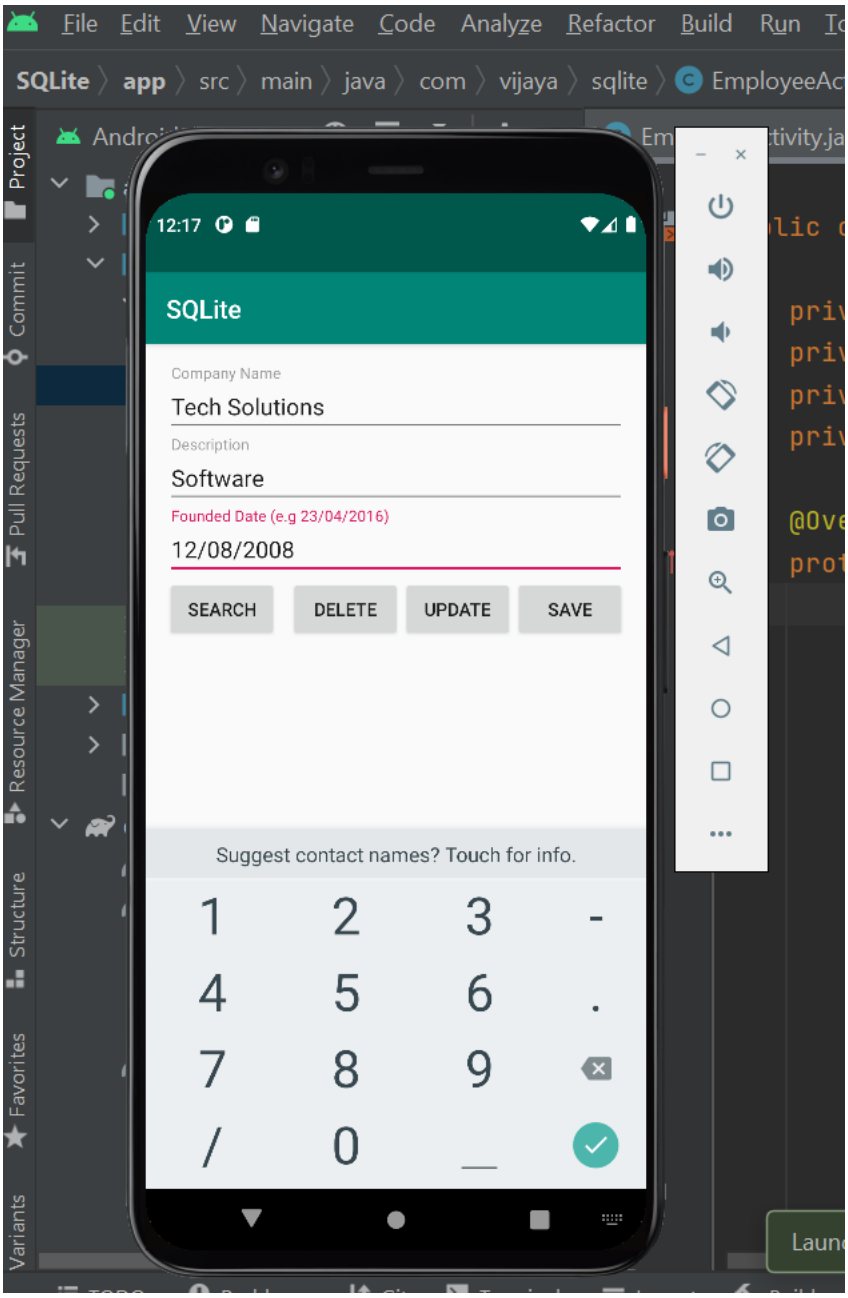
```
<Button
    android:id="@+id/updateButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@id/foundedInputLayout"
    android:layout_toLeftOf="@id/saveButton"
    android:layout_marginBottom="30dp"
    android:text="Update" />

<Button
    android:id="@+id/deleteButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@id/foundedInputLayout"
    android:layout_toLeftOf="@id/updateButton"
    android:layout_marginBottom="30dp"
    android:text="Delete" />
```

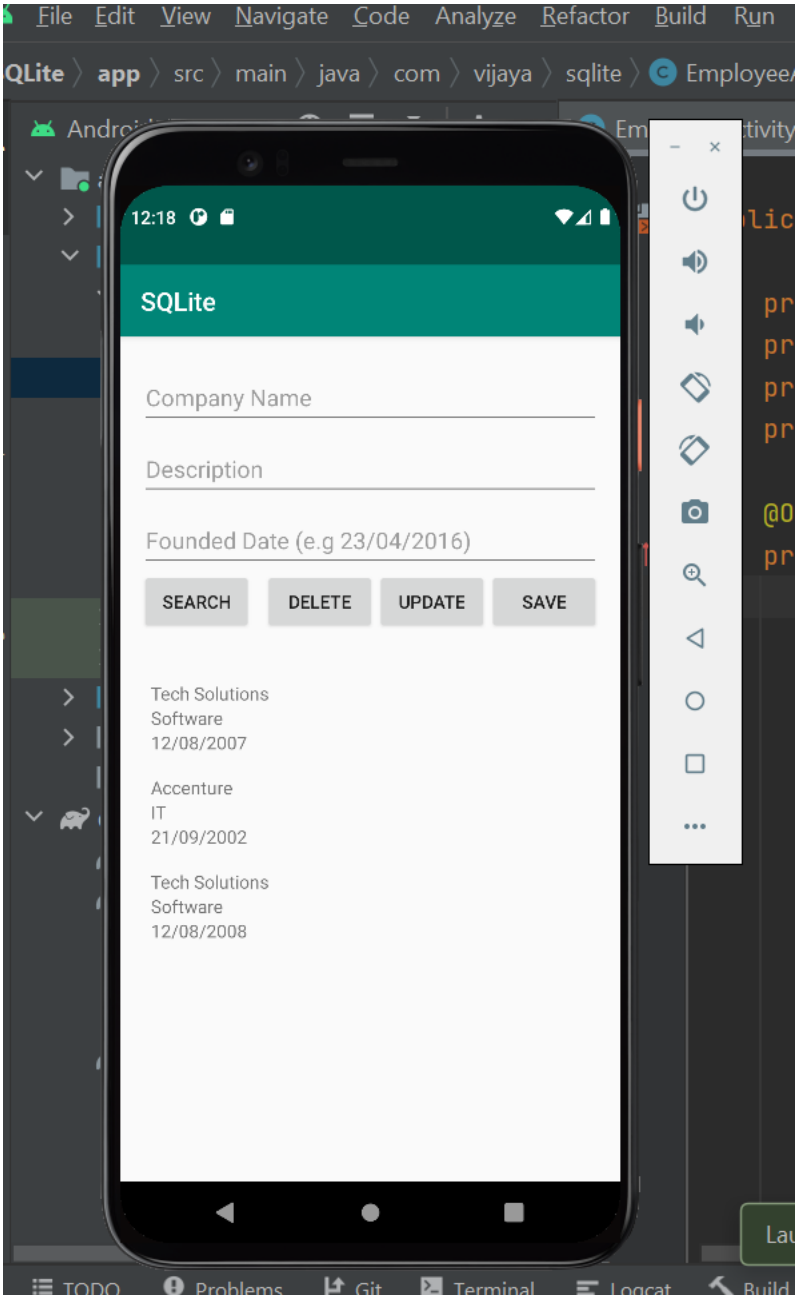
Output:

Employer and employee activity are displayed on the app's home screen.

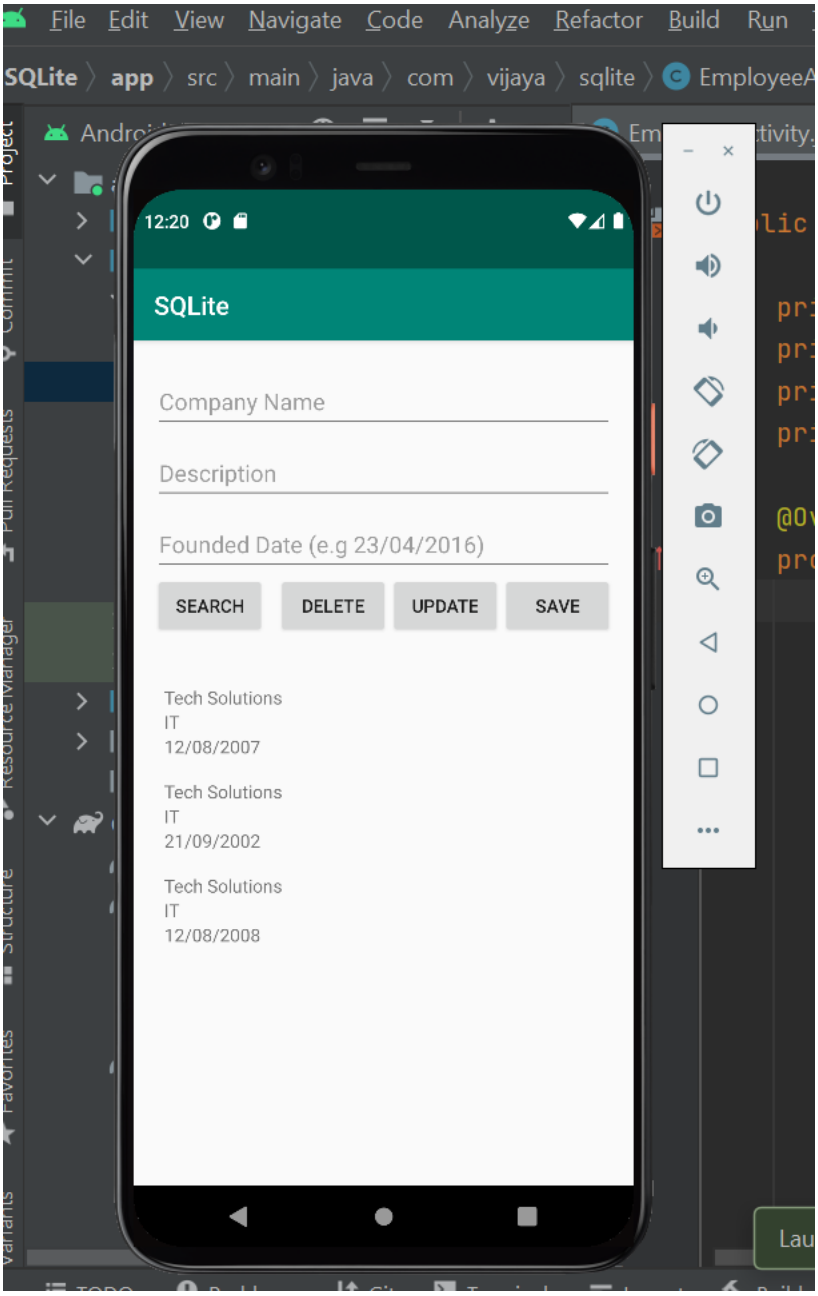
When the user selects the employer button on the home screen, it takes them to the next screen, where they can enter the employer's information.



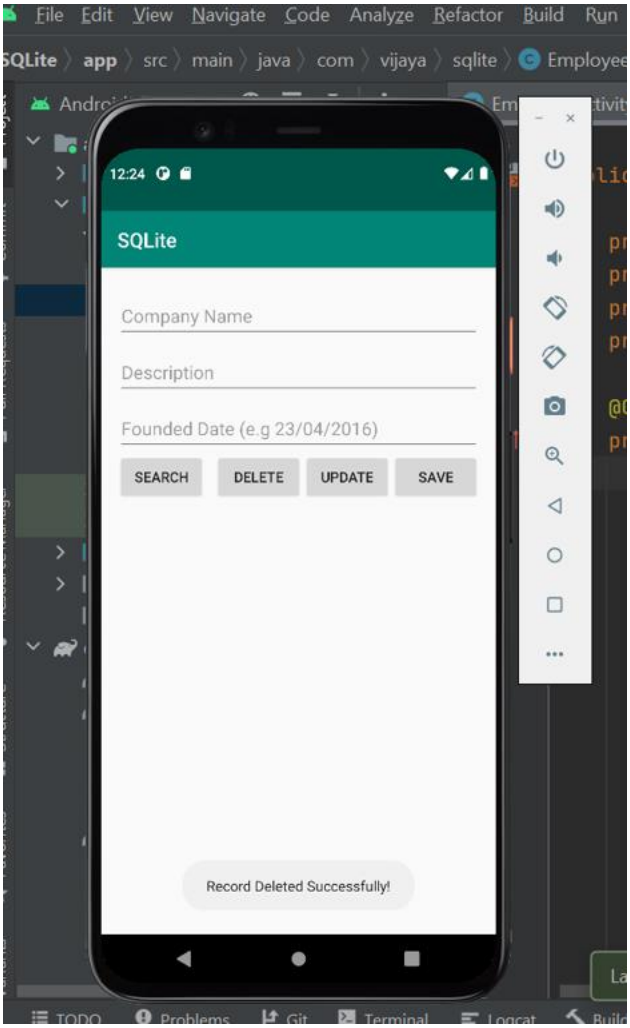
The record is saved when the user clicks the 'SAVE' button, and it is displayed when the user clicks the 'SEARCH' button.



By clicking the 'UPDATE' button, you can make changes to the record information you've entered. The success message can be displayed once the record has been updated, as seen below.



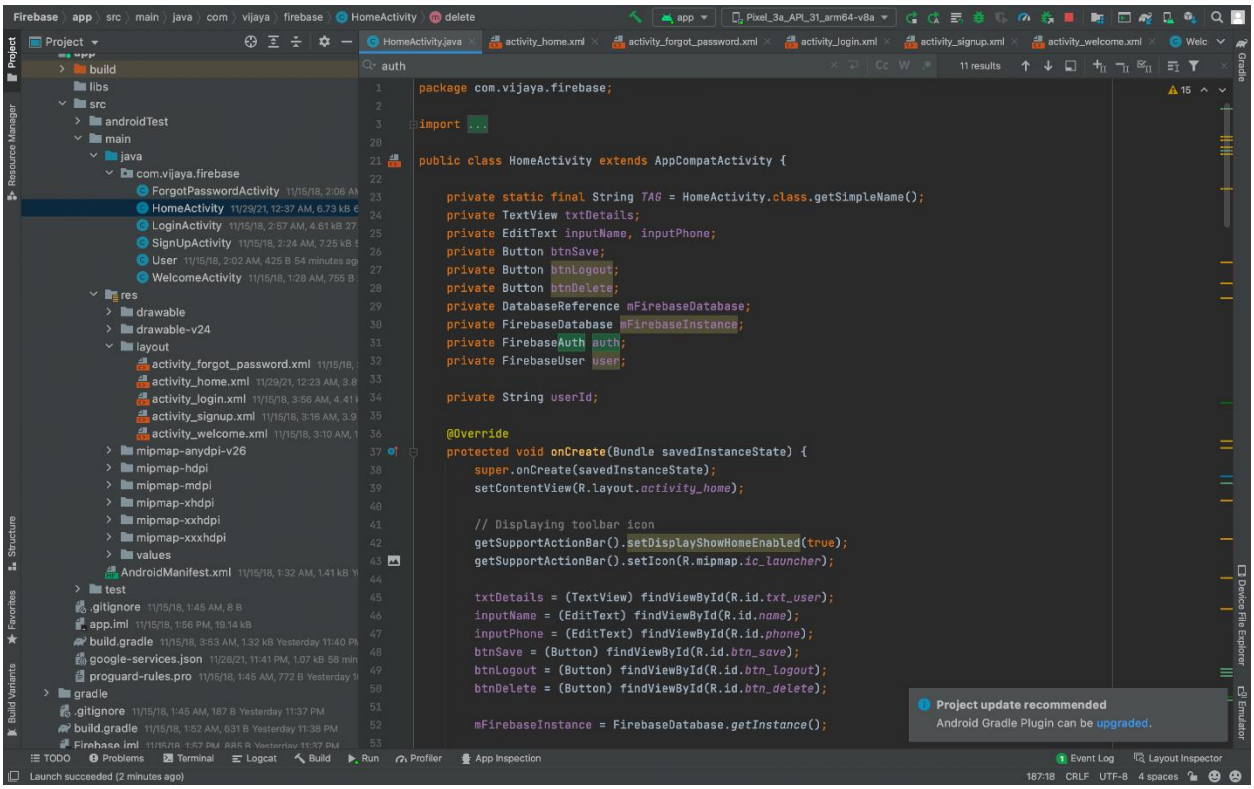
By hitting the 'DELETE' button, the record can be removed. The success message can be displayed after the record has been erased, as seen below.

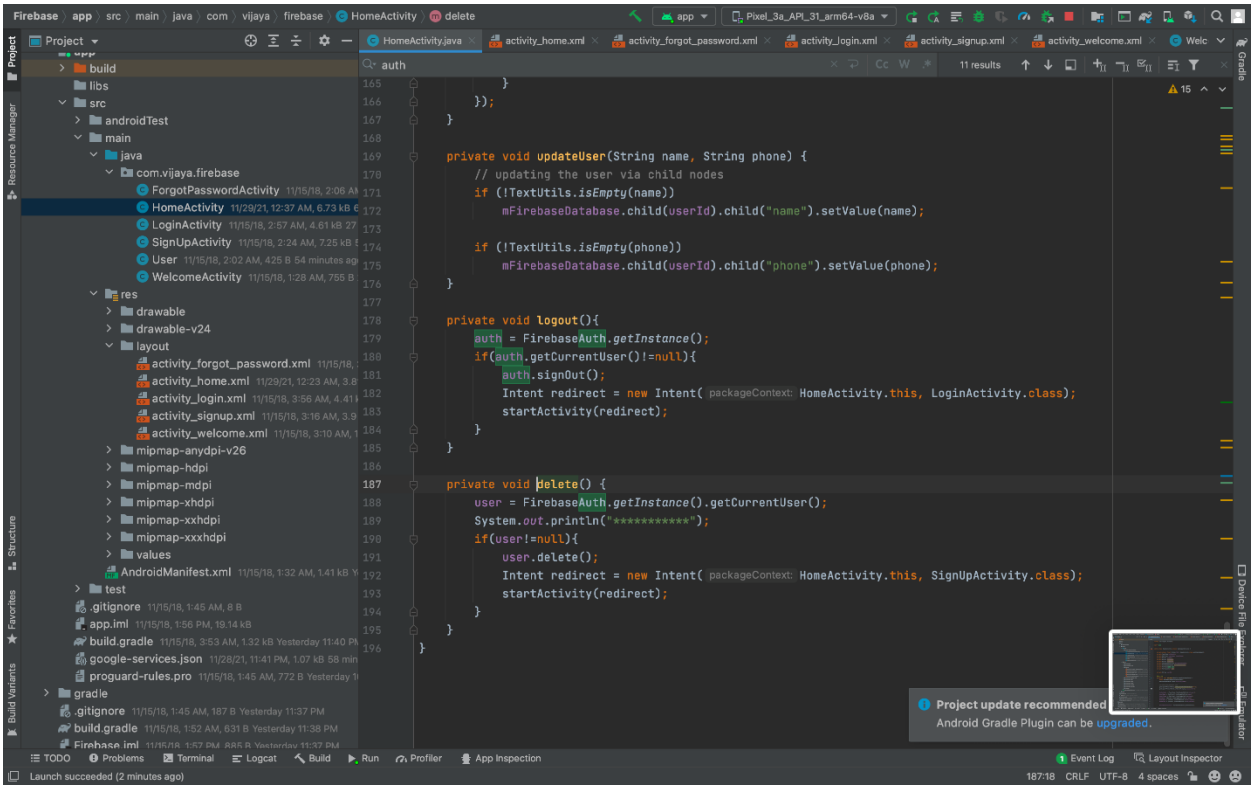


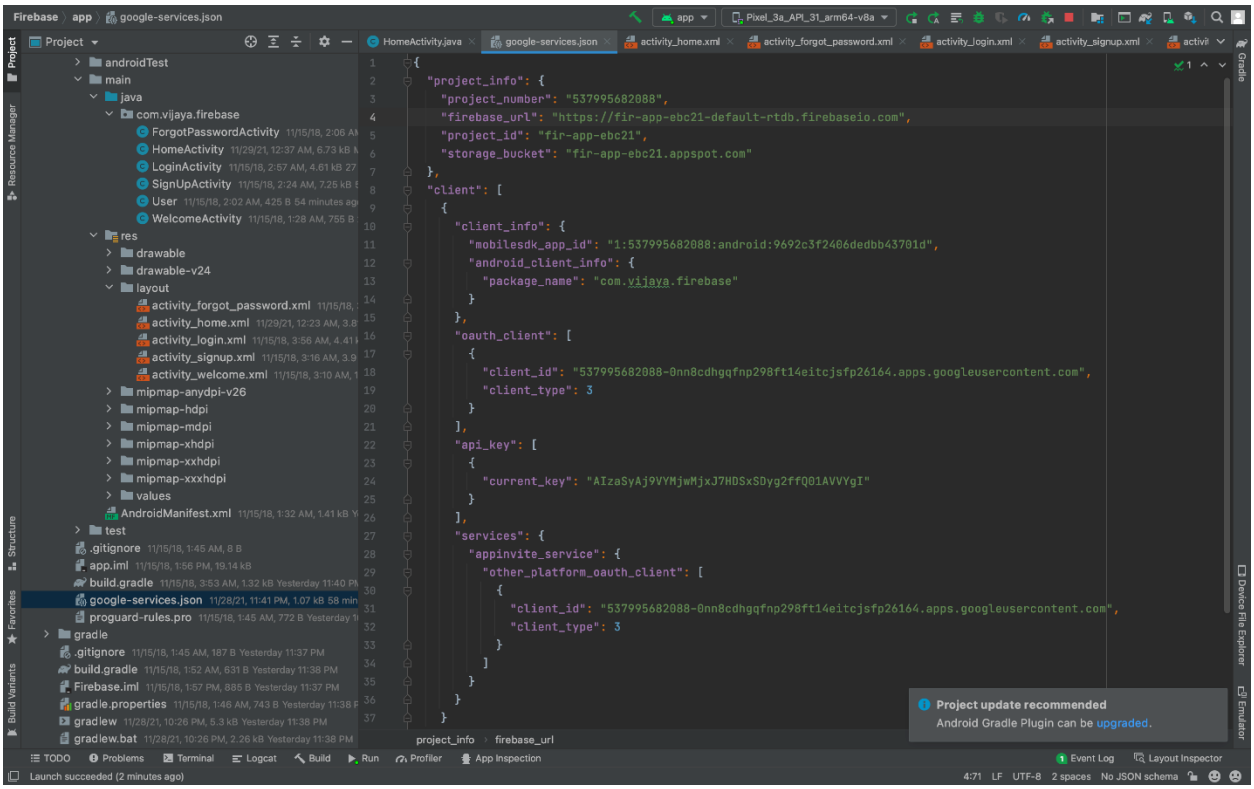
Task 2: Firebase:

To sign up or sign in the user, I created a Firebase application. The application now has the ability to log out and delete user accounts. The following is the source code:

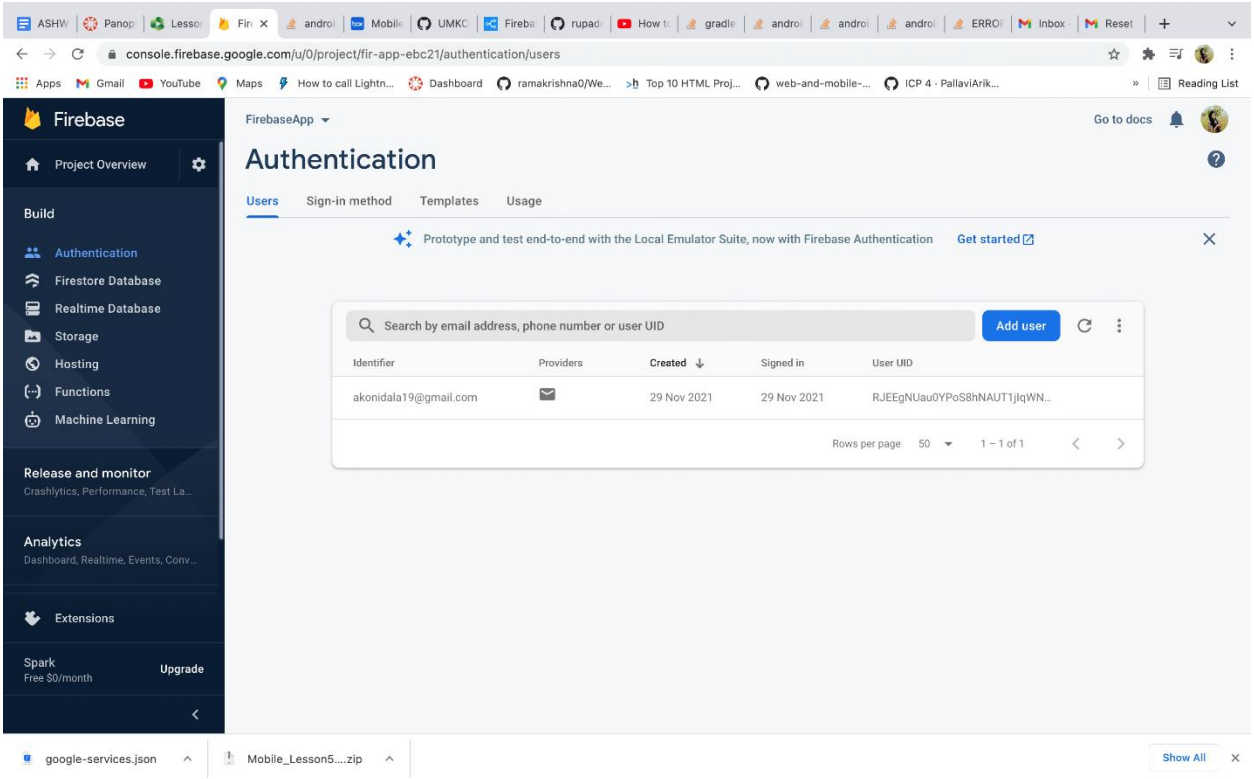
As described below, I downloaded and changed the relevant JSON file (google-services) in the application.

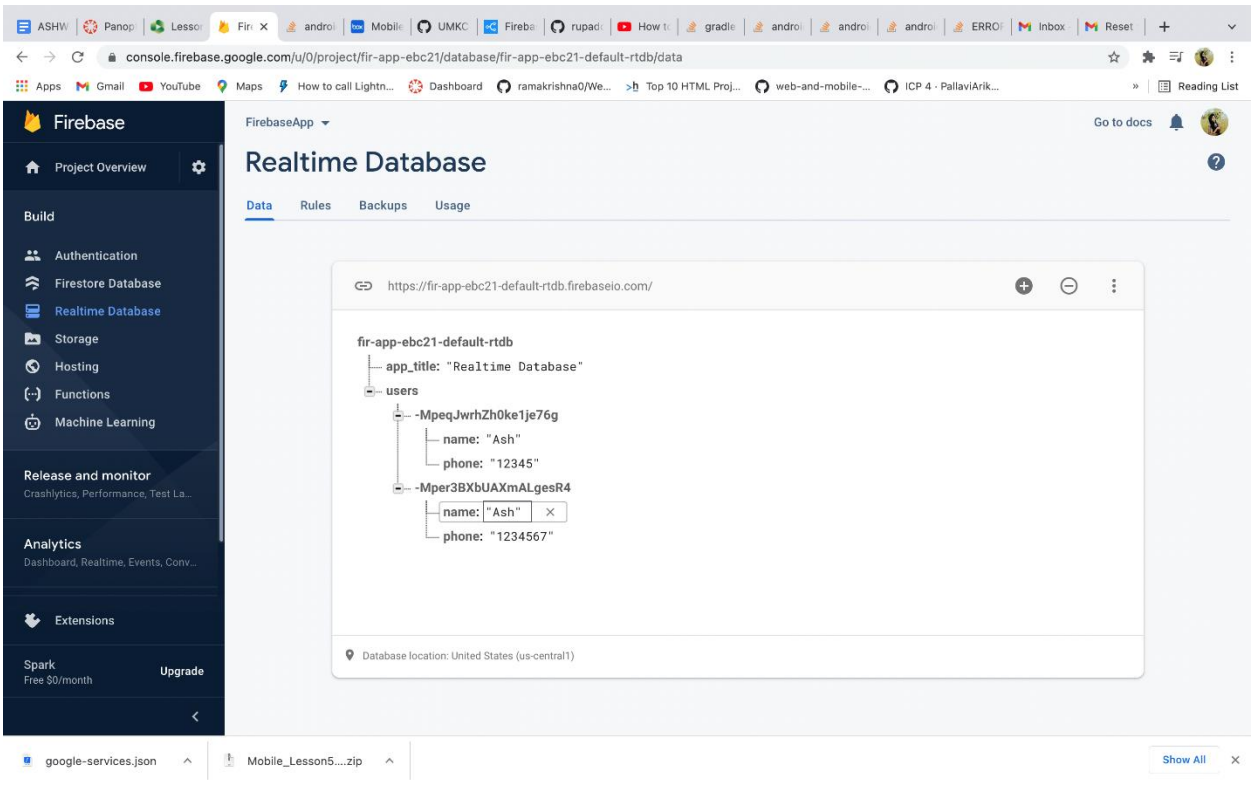

Firebase › app › src › main › java › com › vijaya › firebase › HomeActivity › delete
HomeActivity.java
activity_home.xml
activity_forgot_password.xml
activity_login.xml
activity_signup.xml
activity_welcome.xml
auth
11 results
package com.vijaya.firebase;
import ...
public class HomeActivity extends AppCompatActivity {
private static final String TAG = HomeActivity.class.getSimpleName();
private TextView txtDetails;
private EditText inputName, inputPhone;
private Button btnSave;
private Button btnLogout;
private Button btnDelete;
private DatabaseReference mFirebaseDatabase;
private FirebaseDatabase mFirebaseInstance;
private FirebaseAuth auth;
private FirebaseUser user;
private String userId;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_home);
// Displaying toolbar icon
getSupportActionBar().setDisplayShowHomeEnabled(true);
getSupportActionBar().setIcon(R.mipmap.ic_launcher);
txtDetails = (TextView) findViewById(R.id.txt_user);
inputName = (EditText) findViewById(R.id.name);
inputPhone = (EditText) findViewById(R.id.phone);
btnSave = (Button) findViewById(R.id.btn_save);
btnLogout = (Button) findViewById(R.id.btn_logout);
btnDelete = (Button) findViewById(R.id.btn_delete);
mFirebaseInstance = FirebaseDatabase.getInstance();
Project update recommended
Android Gradle Plugin can be upgraded.
TODO
Problems
Terminal
Logcat
Build
Run
Profiler
App Inspection
Event Log
Layout Inspector
Launch succeeded (2 minutes ago)
187:18
CRLF
UTF-8
4 spaces

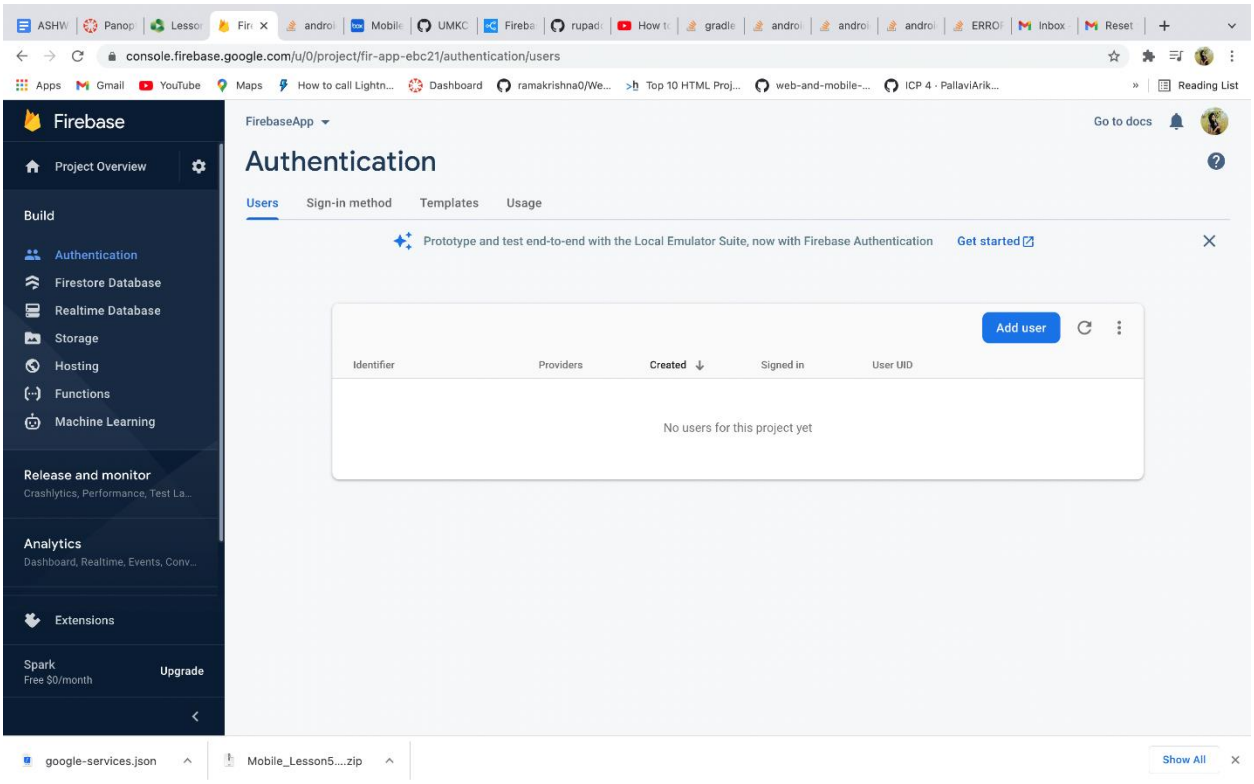

Firebase › app › src › main › java › com › vijaya › firebase › HomeActivity › delete
HomeActivity.java
activity_home.xml
activity_forgot_password.xml
activity_login.xml
activity_signup.xml
activity_welcome.xml
auth
11 results
}
});
}
private void updateUser(String name, String phone) {
// updating the user via child nodes
if (!TextUtils.isEmpty(name))
mFirebaseDatabase.child(userId).child("name").setValue(name);
if (!TextUtils.isEmpty(phone))
mFirebaseDatabase.child(userId).child("phone").setValue(phone);
}
private void logout(){
auth = FirebaseAuth.getInstance();
if(auth.getCurrentUser()!=null){
auth.signOut();
Intent redirect = new Intent(packageContext: HomeActivity.this, LoginActivity.class);
startActivity(redirect);
}
}
private void delete() {
user = FirebaseAuth.getInstance().getCurrentUser();
System.out.println("***********");
if(user!=null){
user.delete();
Intent redirect = new Intent(packageContext: HomeActivity.this, SignUpActivity.class);
startActivity(redirect);
}
}
}
Project update recommended
Android Gradle Plugin can be upgraded.
TODO
Problems
Terminal
Logcat
Build
Run
Profiler
App Inspection
Event Log
Layout Inspector
Launch succeeded (2 minutes ago)
187:18
CRLF
UTF-8
4 spaces



In the firebase, for the authentication sign-in method, Email/password is enabled as shown below.

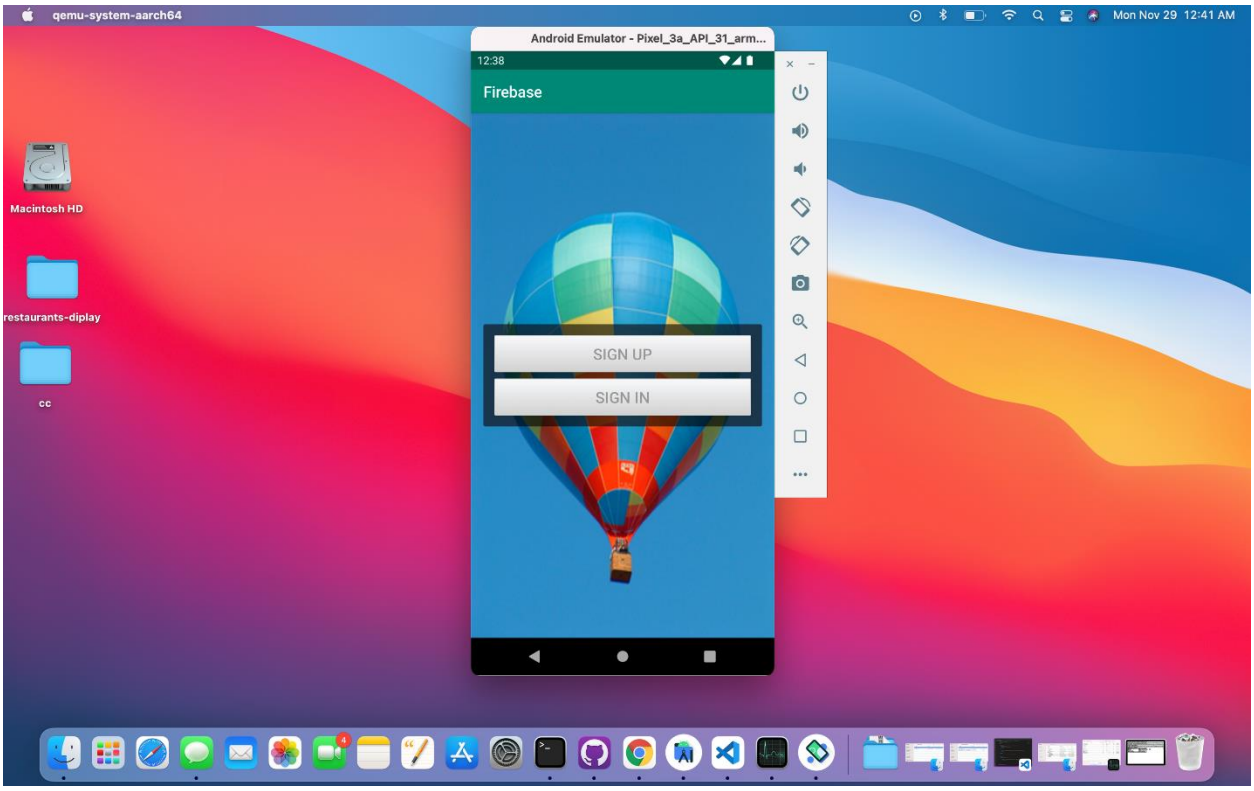



console.firebase.google.com/u/0/project/fir-app-ebc21/database/fir-app-ebc21-default-rtdb/data
Firebase
FirebaseApp
Go to docs
Project Overview
Realtime Database
Data
Rules
Backups
Usage
Build
Authentication
Firestore Database
Realtime Database
Storage
Hosting
Functions
Machine Learning
Release and monitor
Crashlytics, Performance, Test La...
Analytics
Dashboard, Realtime, Events, Conv...
Extensions
Spark
Free $0/month
Upgrade
https://fir-app-ebc21-default-rtdb.firebaseio.com/
fir-app-ebc21-default-rtdb
app_title: "Realtime Database"
users
-MpeqJwrhZh0ke1je76g
name: "Ash"
phone: "12345"
-Mper3BXbUAXmALgesR4
name: "Ash"
phone: "1234567"
Database location: United States (us-central1)
google-services.json
Mobile_Lesson5....zip
Show All

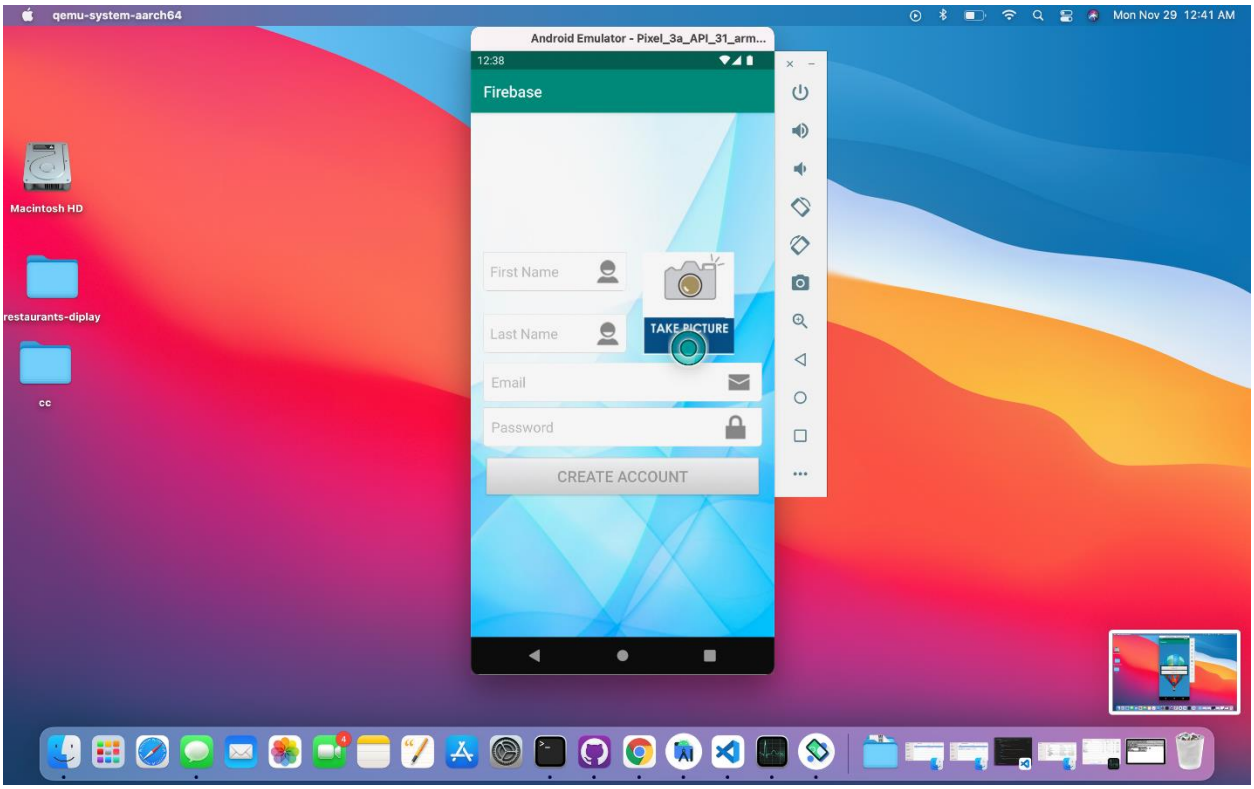

console.firebase.google.com/u/0/project/fir-app-ebc21/authentication/users
Firebase
FirebaseApp
Go to docs
Project Overview
Authentication
Users
Sign-in method
Templates
Usage
Prototype and test end-to-end with the Local Emulator Suite, now with Firebase Authentication
Get started
Build
Authentication
Firestore Database
Realtime Database
Storage
Hosting
Functions
Machine Learning
Release and monitor
Crashlytics, Performance, Test La...
Analytics
Dashboard, Realtime, Events, Conv...
Extensions
Spark
Free $0/month
Upgrade
Add user
Identifier
Providers
Created
Signed in
User UID
No users for this project yet
google-services.json
Mobile_Lesson5....zip
Show All

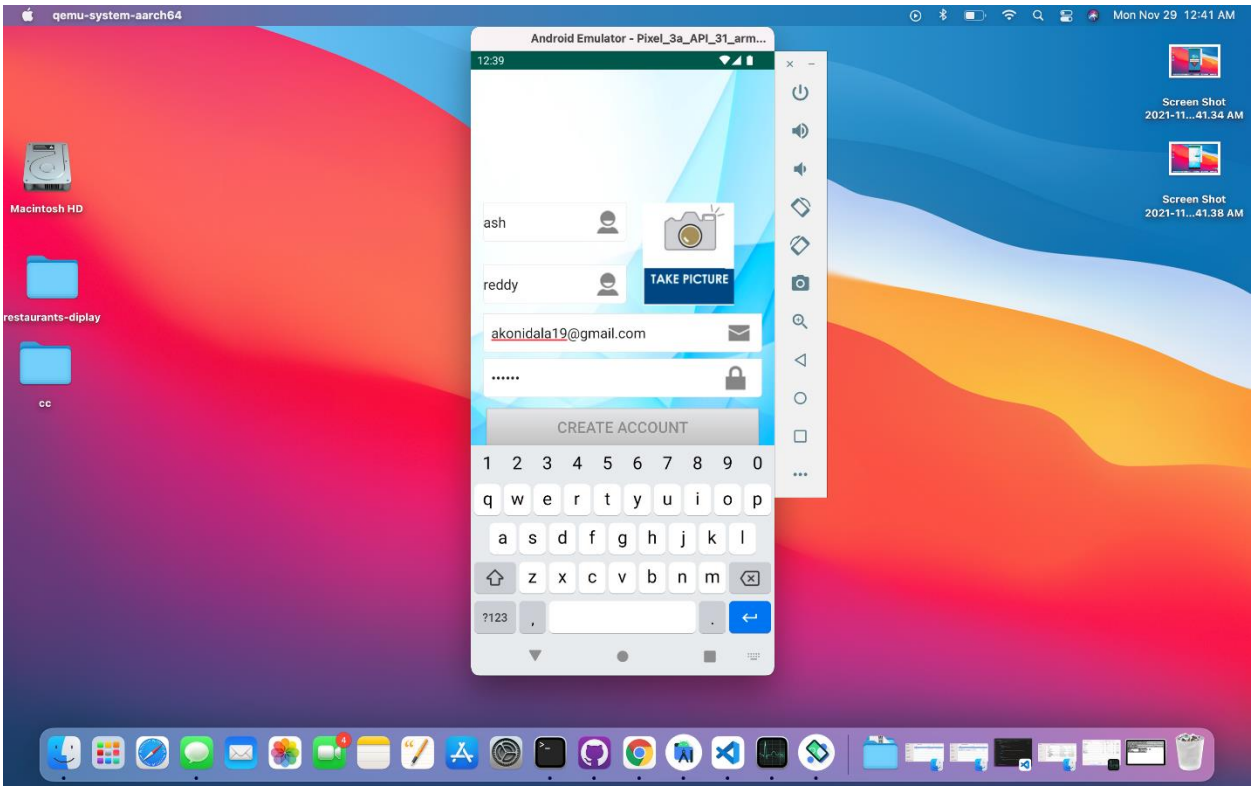
Output:

The home screen of the application is as shown below.

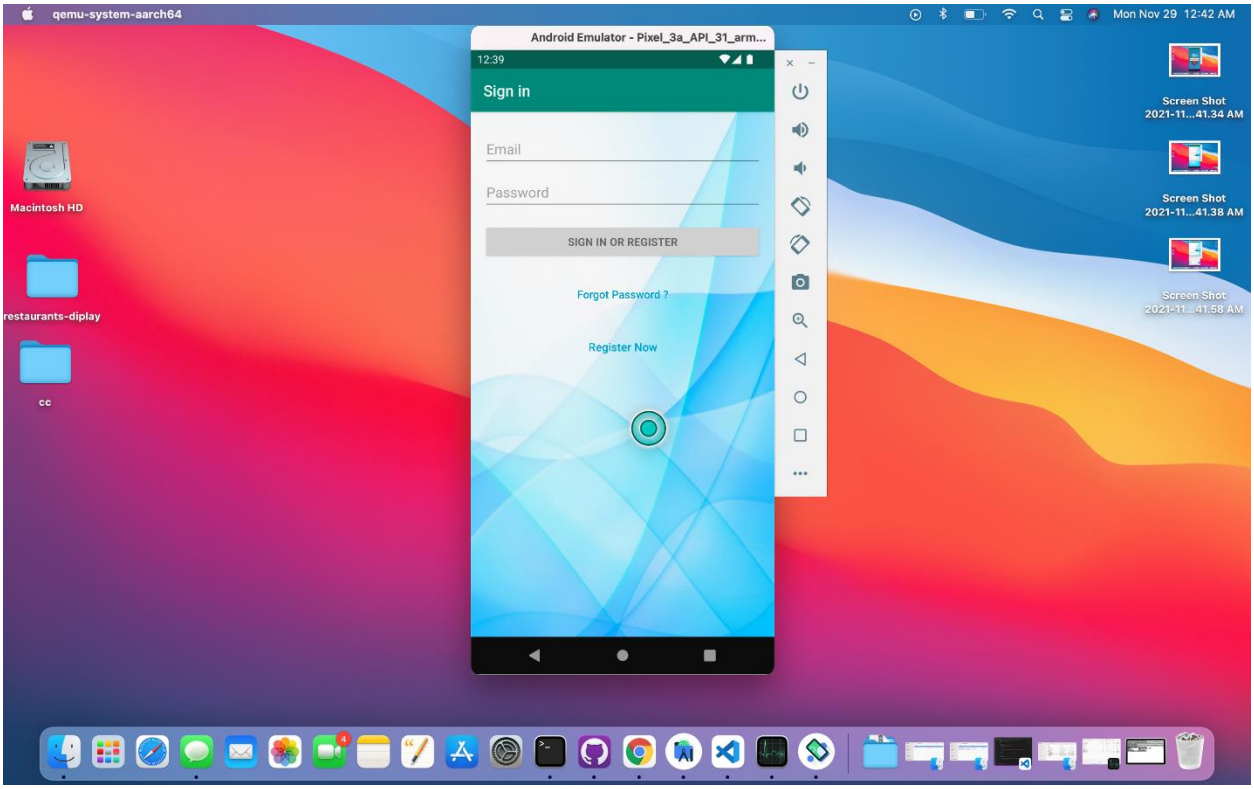


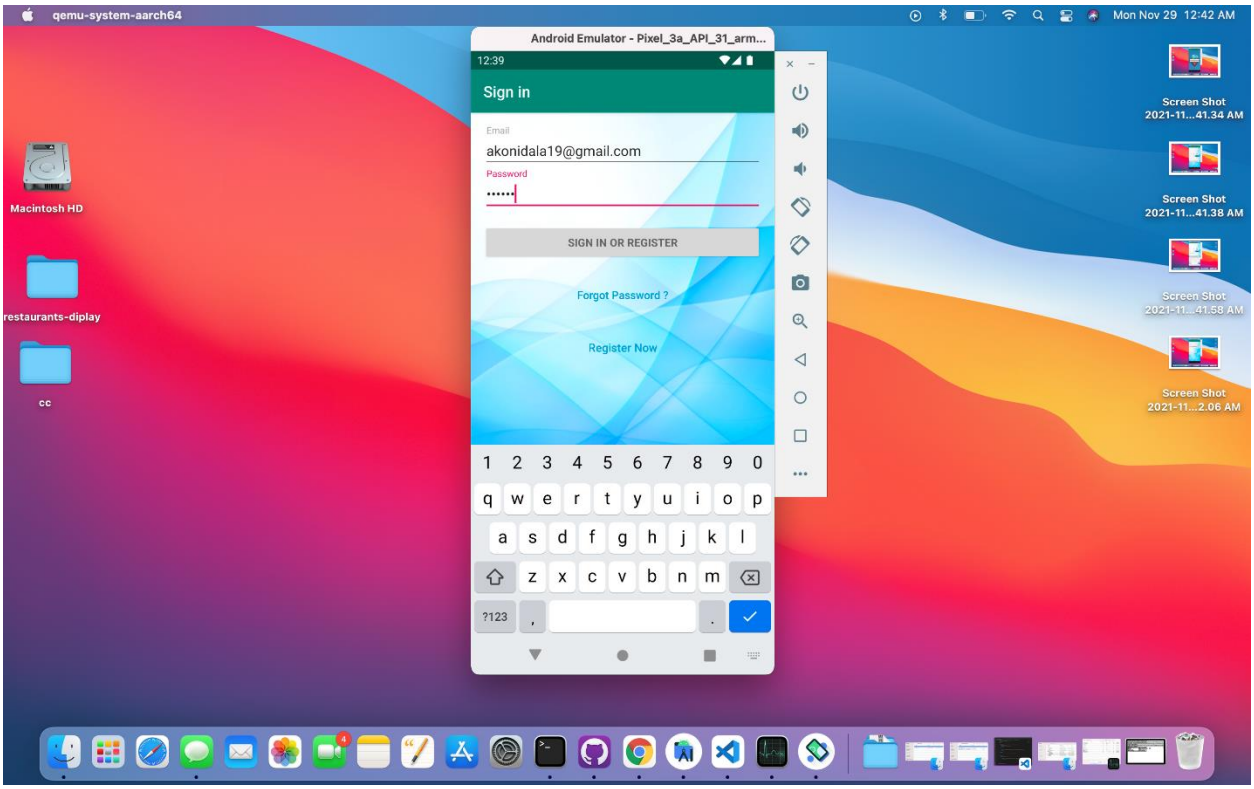
When the user selects the sign-up option on the home screen, it takes them to the next screen, where they may fill out their information and click the 'CREATE ACCOUNT' button. When you refresh the page, the user is updated in the Firebase Authentication users.


qemu-system-aarch64
Mon Nov 29 12:41 AM
Android Emulator - Pixel_3a_API_31_arm...
12:38
Firebase
First Name
Last Name
TAKE PICTURE
Email
Password
CREATE ACCOUNT
Macintosh HD
restaurants-diplay
cc

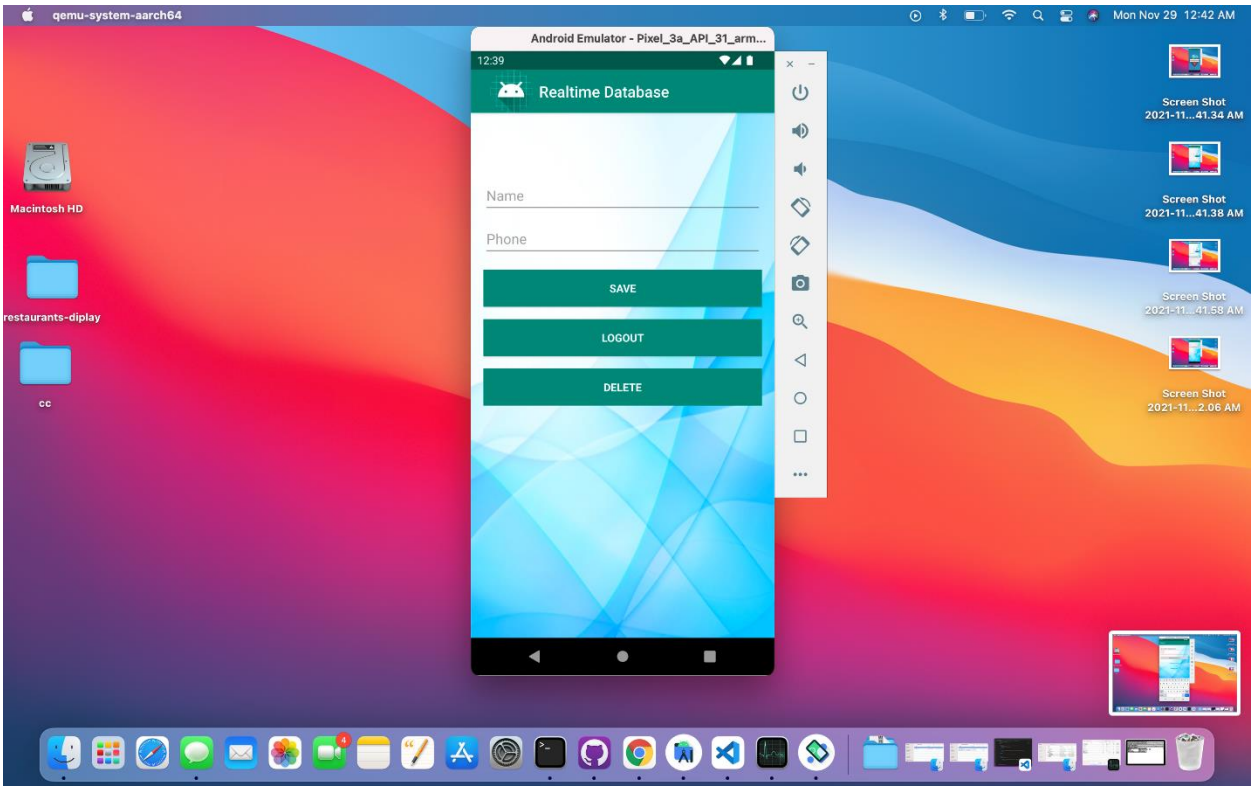

qemu-system-aarch64
Mon Nov 29 12:41 AM
Android Emulator - Pixel_3a_API_31_arm...
12:39
ash
reddy
TAKE PICTURE
akonidala19@gmail.com
CREATE ACCOUNT
Screen Shot 2021-11...41.34 AM
Screen Shot 2021-11...41.38 AM
Macintosh HD
restaurants-diplay
cc

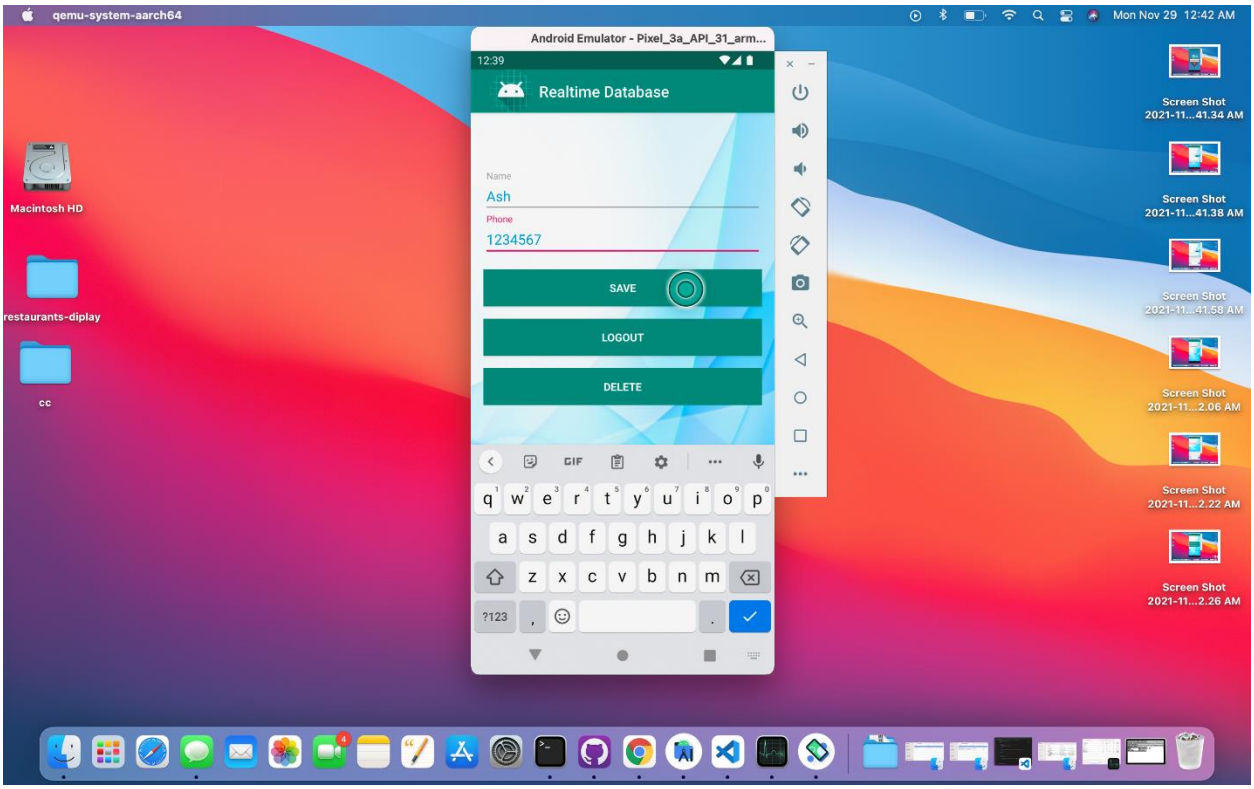
When the user clicks the 'SIGN IN' button on the home screen, it takes them to the next screen, where they can fill out their information and click the 'SIGN IN OR REGISTER' button.





Enter the name and phone and click on 'SAVE' button.




qemu-system-aarch64
Mon Nov 29 12:42 AM
Android Emulator - Pixel_3a_API_31_arm...
12:39
Realtime Database
Name
Ash
Phone
1234567
SAVE
LOGOUT
DELETE
Macintosh HD
restaurants-diplay
cc
Screen Shot 2021-11...41.34 AM
Screen Shot 2021-11...41.38 AM
Screen Shot 2021-11...41.58 AM
Screen Shot 2021-11...2.06 AM
Screen Shot 2021-11...2.22 AM
Screen Shot 2021-11...2.26 AM