Summer Project On

Customer2ContarctorServices

By

Amit Manohar Rathod (2021510053)

Under the guidance of

Internal Supervisor

Dr. AARTI M. KARANDE

Department of Master Of Computer Application
Sardar Patel Institute of Technology
Autonomous Institute Affiliated to Mumbai University

2022-23

CERTIFICATE OF APPROVAL

This is to certify that the following students

Amit Manohar Rathod (2021510053)

Have satisfactorily carried out work on the project entitled

"Customer2ContarctorServices"

Towards the fulfillment of the project, as laid down
by
Sardar Patel Institute of Technology
during year
2022-23.

Project Guide:
Dr. AARTI M. KARANDE

PROJECT APPROVAL CERTIFICATE

This is to certify that the following students

Amit Manohar Rathod (2021510053)

Have successfully completed the Project report on

"Customer2ContarctorServices",

which is found to be satisfactory and is approved

at

SARDAR PATEL INSTITUTE OF TECHNOLOGY,
ANDHERI (W), MUMBAI

INTERNAL EXAMINER EXTERNAL EXAMINER

HEAD OF DEPARTMENT PRINCIPAL

Contents

Abstract

People who want to renovate their homes, need any Interior to design their house, or even as simple as getting any maintenance service like getting their home appliances connected to the inverter, etc. find it challenging to reach out to the Service Providers. This is the scenario where our application comes into the picture.

Sometimes Clients also face the problem of searching for a professional and experienced electrician at a particular time. So, to overcome this kind of problem, an on-demand C2C app can be launched by which people can contact the professionals from their feed provided on the home page.

This Application can be used by clients who cannot easily reach out to professionals as well as Professionals who have fewer opportunities in the market, especially after the pandemic.

Objectives

The Android-based Application "Customer2ContarctorServices" is used

- To solve some loopholes in the offline business by making it online and bringing the connectivity.
- To Establish a connection between Clients and Contractors.
- To break the Monopoly in the local areas. Get more players in the market.
- Providing Clients maintenance services at fingertips.

List of Figures

List of Tables

1 Introduction

1.1 Problem Definition

To design an android application where user can find different variety of workers for their home furnishings, decoration, makeover and provide service providers better opportunities and scope.

1.2 Objectives and Scope

1.2.1 Objectives

The Android-based Application "Customer2ContarctorServices" is used

- To solve some loopholes in the offline business by making it online and bringing connectivity.
- To Establish a connection between Clients and Contractors.
- To break the Monopoly in the local areas. Get more players in the market.
- Providing Clients maintenance services at fingertips.

1.2.2 Scope

The customer can provide his/her details in the profile and view various contractors and their work on the app.

In the application, the Contractors must enter their profession and other details in the profile section.

Our System is being made for reducing the efforts and struggles customers have while choosing the right person/contractors for the job they needed to complete. The communication between Customers and contractors can happen through the contact details they have provided in their profile while signing up.

1.3 Existing System

Currently, an application exists like JustDial for hiring workers for doing whatever tasks you like.
Some of the disadvantages of existing system are as follows :

- Distracting
  Every time you open an application or website it will show you too many things to choose from besides normal household decor and repair things.
- Time-consuming
  You have to browse through multiple tabs to get to the service provider's window.

1.4 Proposed System

The User can be any person they can register themselves by email and can have access to system features. This system will be showing different contractors, with different professions on the home tab.

The customer can add their request or type of work they need to done and contractors can have look , at what they are.

You can also look at what kind of work the contractors have done in the past.

Some of the advantages of our system are as follows :

- User Friendly
  It provides an attractive interface to the user for navigation through a dynamic flow of the contractor's profile and details.
- Simple Communication
  Users can upload their own requirements and contractors can contact them or customers can check the work of contractors and contact them.

1.5 System Requirements

- Hardware Requirements on Server Side

Table 1.5.1: Hardware Requirements on Server Side

Processor	Dual Core Processor or Above
RAM	Minimum 4 GB RAM
Storage	Minimum 10 GB Hard Disk Space for smooth run

- Hardware Requirements on Client Side

Table 1.5.2: Hardware Requirements on Client Side

Device	Android Device with Touch Screen minimum 5” inch Display
Processor	Dual Core Processor or Above
RAM	Minimum 2 GB RAM
Storage	Minimum 250 MB Storage Space

- Software Requirements on Server Side

Table 1.5.3: Software Requirements on Server Side

Operating System	OS Independent
Database	Firestore

- Software Requirements on Client Side

Table 1.5.3: Software Requirements on Client Side

Operating System	Android/IOS Smartphone
Server	Not Required

2 Software Requirement Specification (SRS) and Design

2.1 Purpose

The purpose of our project is to develop a UI application that can help users to reach out Service Providers and vice versa will be remarkable. A Service Provider can also list down their services along with charges or packages. This way contractors can have an idea of whom they are working with and what their requirements are.

2.2 Definition

To build a Customer2ContarctorServices Application so the customers and clients can have an easy reach to person suitable for the job.

2.3 Overall Description

2.3.1 Product Functions

The product function includes:

1. View Feed: This consists of the feed where a list of different Service Providers is displayed for the Client in which they can view the profile of the service provider and a list of required Services (User Posts) is displayed in the Service Provider's feed.

2. Posts: Here the user can post to get a particular service which will be visible to the appropriate authority.

3. Uploads: Here Service providers can upload the services that they provide along with the images of work that can be viewed by clients. These uploaded work / Services will be available in the uploads section and in the profile as well.

4. Profile: Client can view their profile and also add the details by setting up their account. Service Provider can also view their profile will all their details of the shop, experience, and work gallery, and also they can setup an account if they haven't done it.

5. From the Feed, both parties can contact each other for further details of services they want.

6. Clients can also search for a particular service from the search view. In search view, a user can search by service they want.

2.3.2 User Characteristics

There are two types of users:

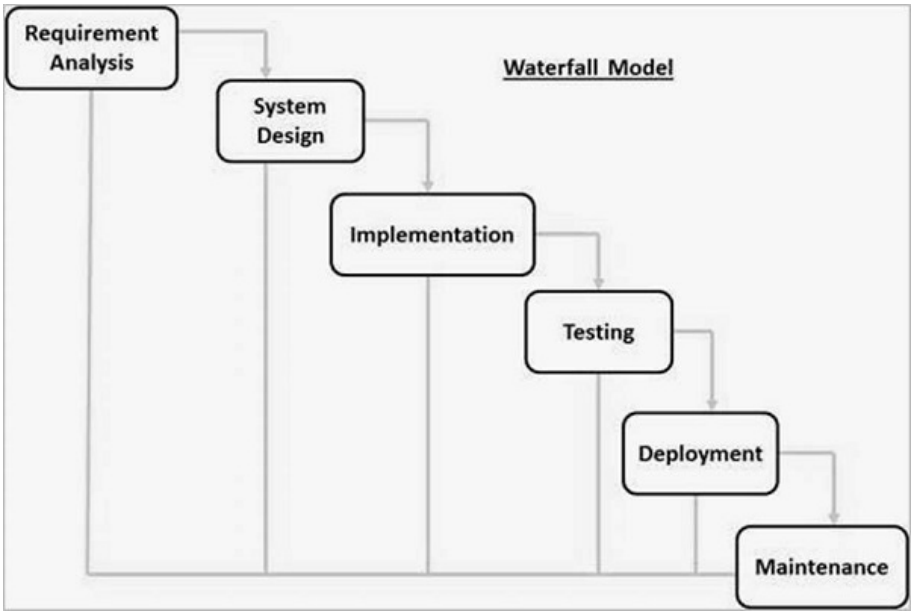
- Customers: Once a user registers themselves then they can have access to contractors and find suitable persons for the tasks needed to be done.
- Contractors: After registration and filling in the details about their shop and the services they provide they have to look on the different requests added by different customers.

3 Project Analysis and Design

3.1 Methodologies Adapted

In Waterfall model, very less customer interaction is involved during the development of the product. Once the product is ready then only it can be demonstrated to the end users.

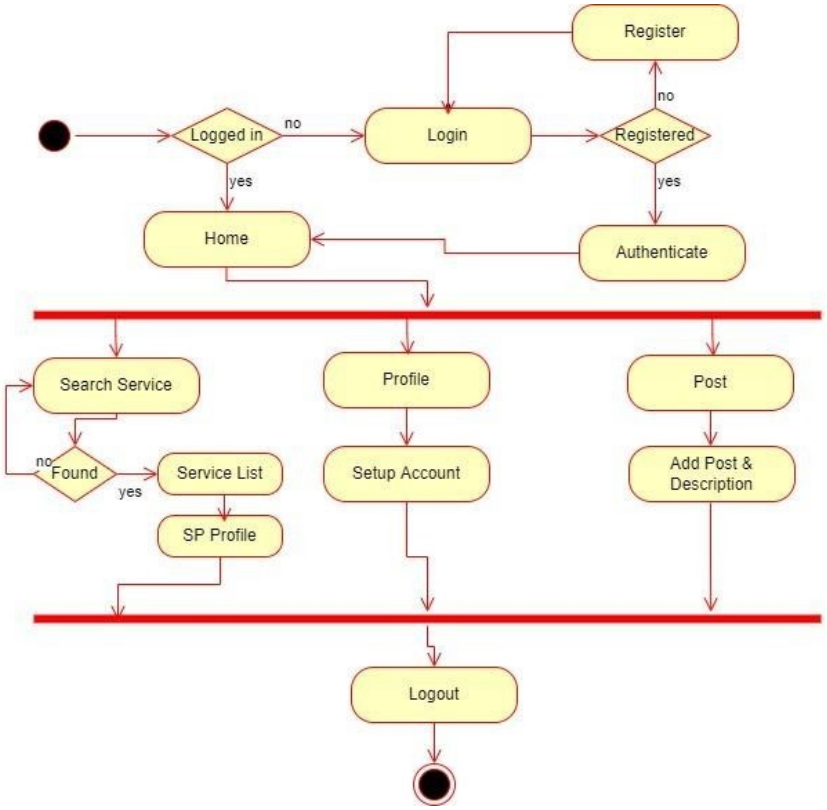
Once the product is developed and if any failure occurs then the cost of such issues is very high, because we need to update everything from document till the logic.



3.1.1: Diagrammatic Representation of Waterfall Model

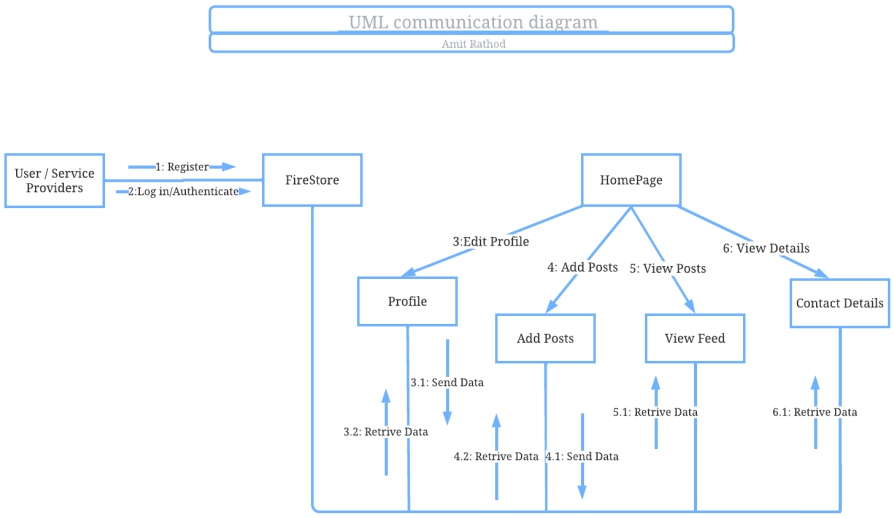
3.2 Modules

3.2.1 Activity diagram



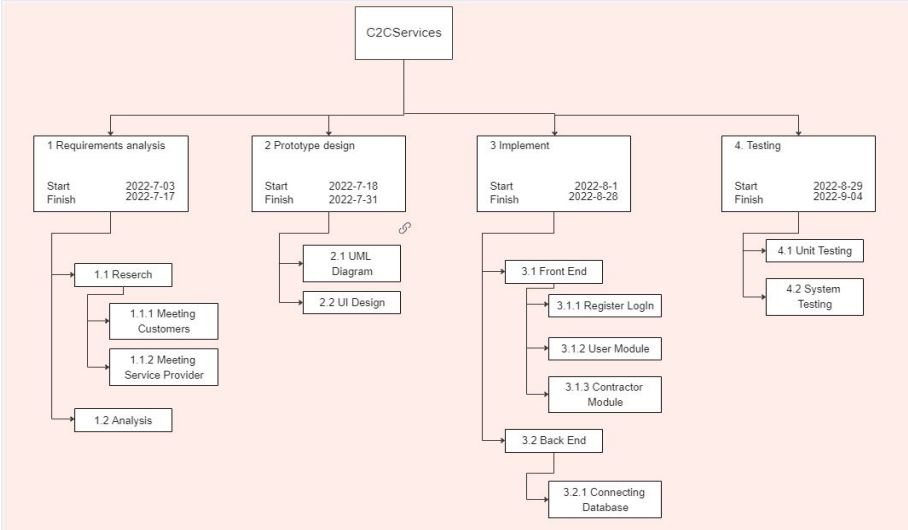
3.2.1: Activity Diagram

3.2.2 Communication Design



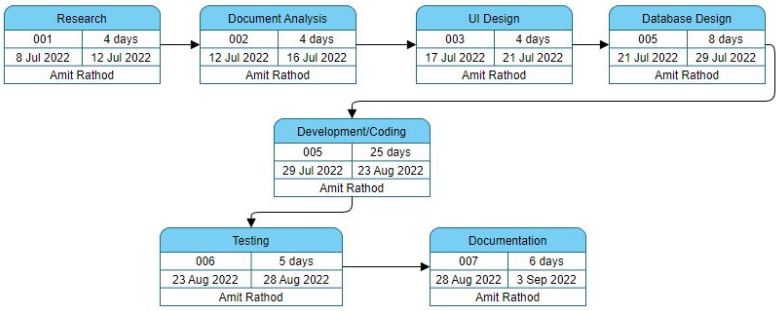
3.2.2: Communication Diagram

3.2.3 Work Breakdown Structure



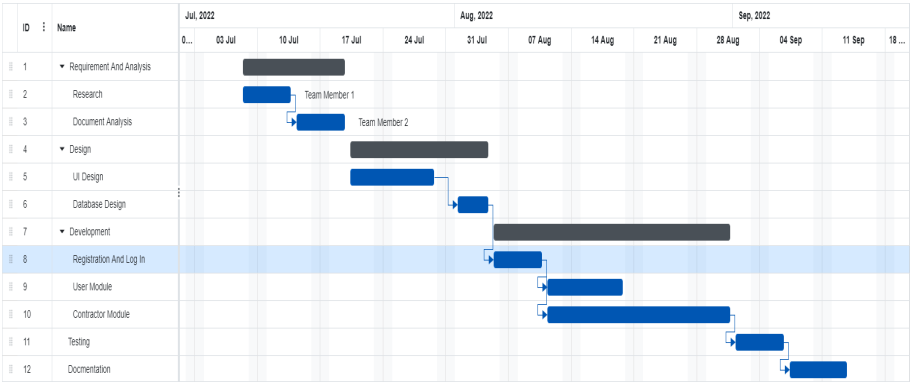
3.2.3: Work Breakdown Structure

3.2.4 PERT Chart



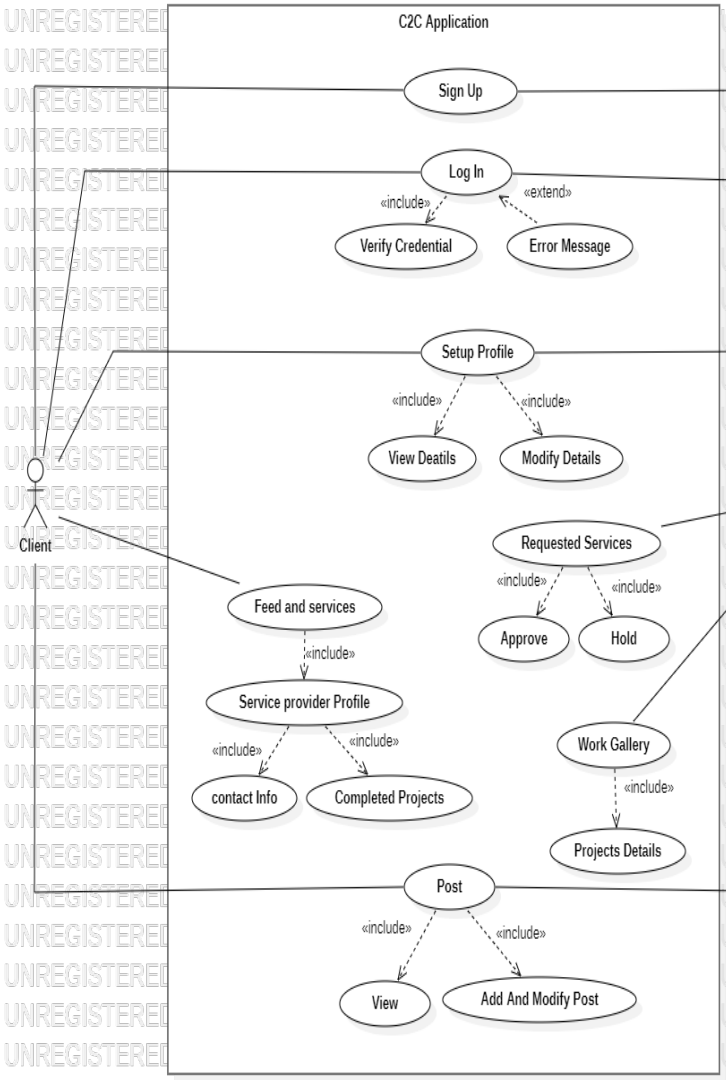
3.2.4: PERT Chart

3.2.5 Gantt Chart



3.2.5: Gantt Chart

3.2.6 Use-Case



3.2.6: Use-Case Diagram

Use Cases:

1. Register
2. Login
3. Update/Edit Profile
4. View Posts and View Requests/work description
5. View Profile and View Work Gallery
6. Add Posts
7. Contact Service Provider

Table 4.2.1: Use Case Table - Register

Use Case ID	1
Use Case Name	Register
Actor	Customer or Contractor
Pre-Condition	They must register themselves first
Post-Condition	User can login
Flow of events	Login,Register or Setup Profile

Table 4.2.2: Use Case Table - Login

Use Case ID	2
Use Case Name	Login
Actor	Customer or Contractor
Pre-Condition	They must register themselves first
Post-Condition	User can view feed posts and profile, shop details
Flow of events	Login,Register or Edit Profile and view posts

Table 4.2.3: Use Case Table - Update Profile

Use Case ID	3
Use Case Name	Update Profile
Actor	Customer or Contractor
Pre-Condition	Login
Post-Condition	User can view or edit the profile

Table 4.2.4: Use Case Table - View Posts

Use Case ID	4
Use Case Name	View Posts and View Requests/work description
Actor	Customer and Contractor
Pre-Condition	Login
Post-Condition	User can view posts and profile details

Table 4.2.5: Use Case Table - View Profile

Use Case ID	5
Use Case Name	View Profile and View Work Gallery
Actor	Customer or Contractor
Pre-Condition	Login
Post-Condition	Can view profile and other details

Table 4.2.6: Use Case Table - Add Posts

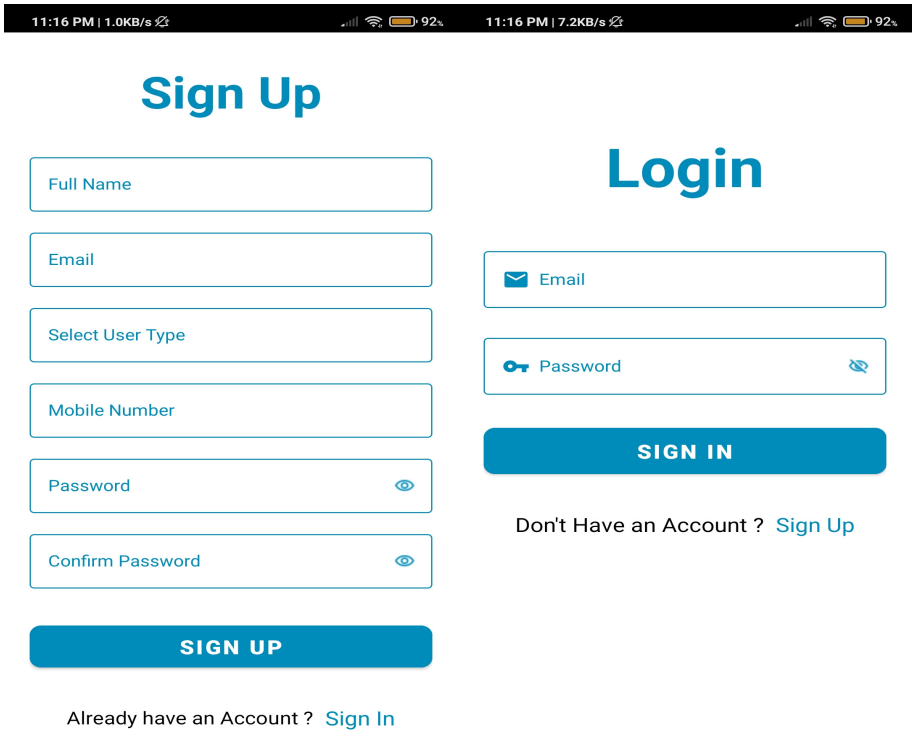
Use Case ID	6
Use Case Name	Add Posts
Actor	Customer or Contractor
Pre-Condition	Login
Post-Condition	Can add posts or work description

Table 4.2.7: Use Case Table - View Contact Details

Use Case ID	7
Use Case Name	Contact Details
Actor	TPC
Pre-Condition	Login
Post-Condition	Can view Contact details of customer/contractor.

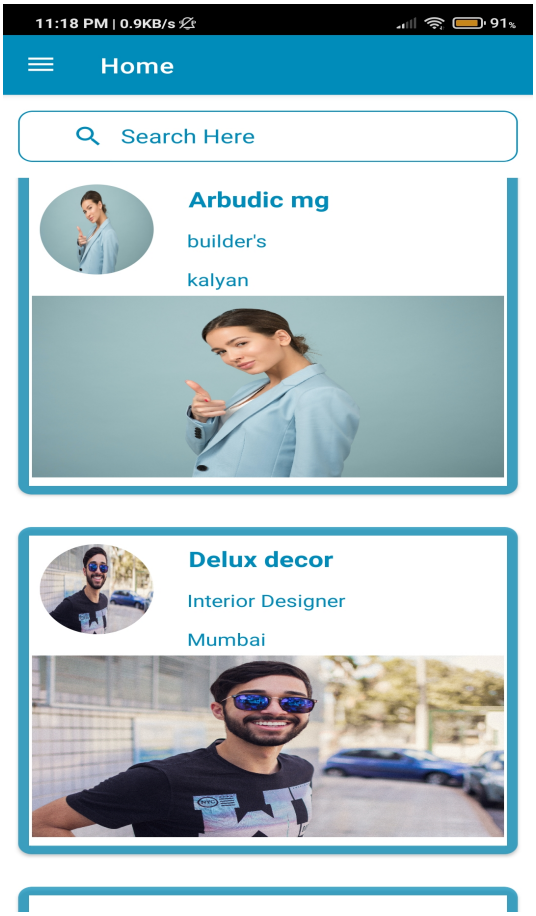
4 Project Implementation and Testing

4.1 Login and Register



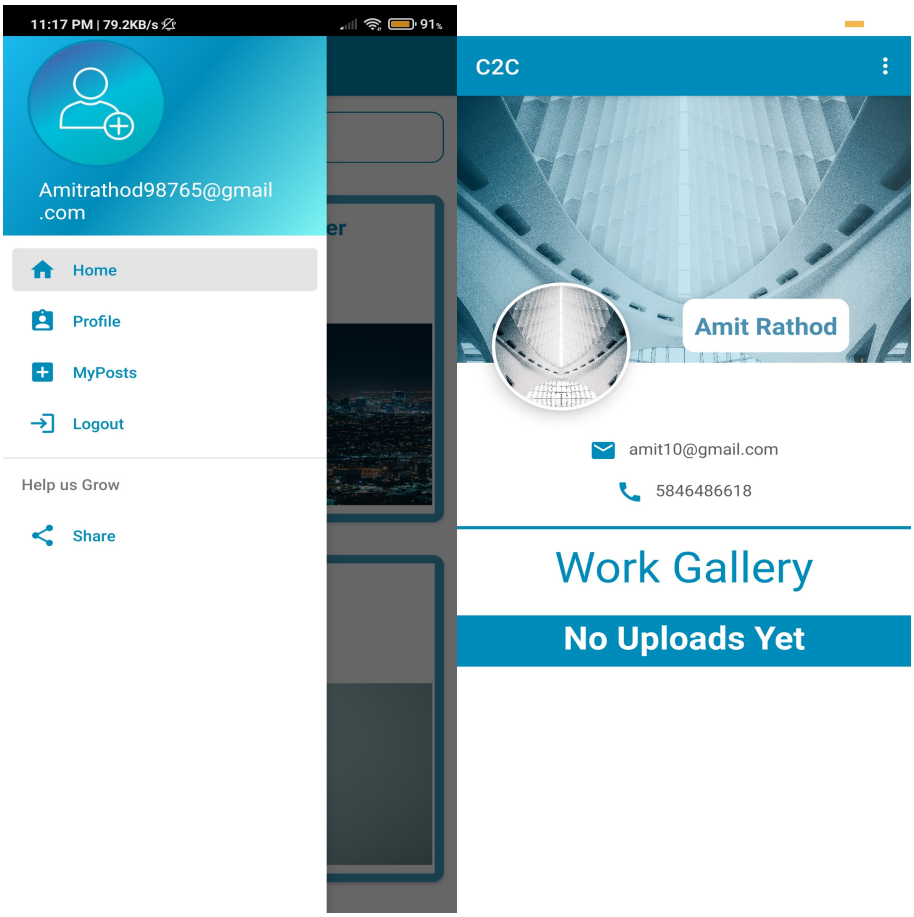
4.1.1: Login and Register

4.2 User Home Page - Landing View



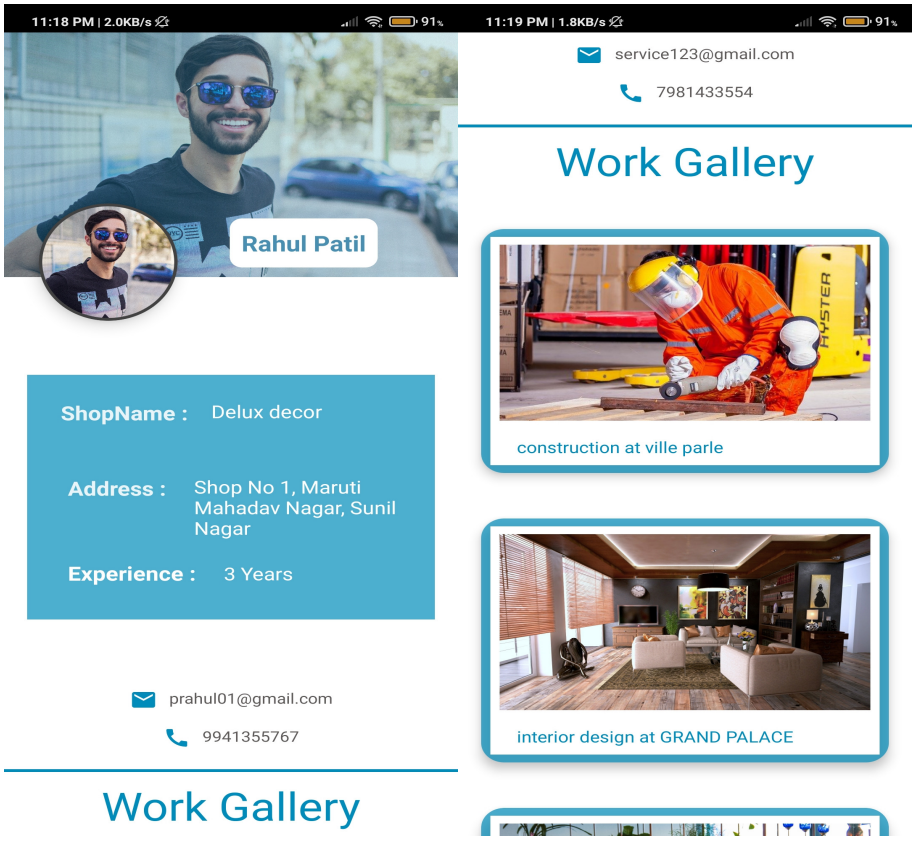
4.2.1: Home View

4.3 Profile



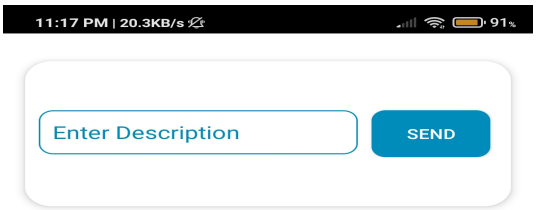
4.3.1: Profile

4.4 Checking Service Provider profile



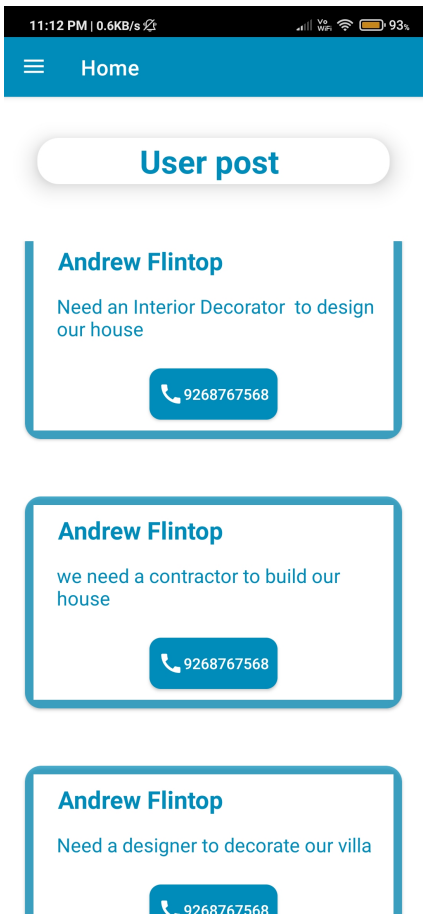
4.4.1: Contractor Profile

4.5 Add Post



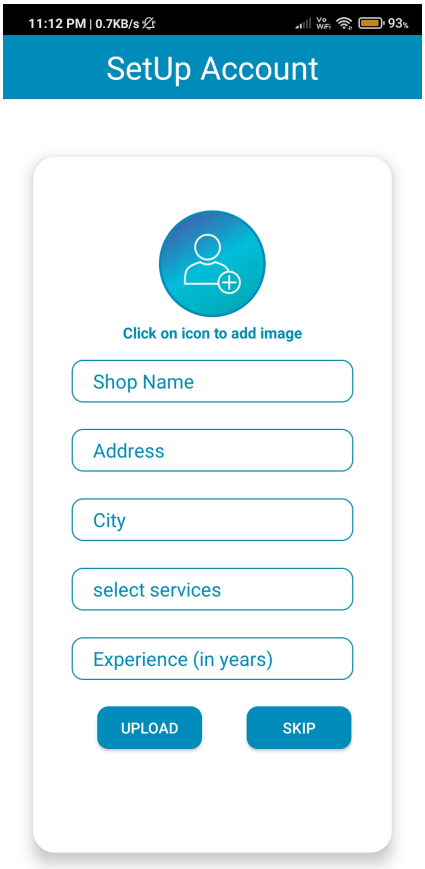
4.5.1: User Adding Post

4.6 Contractor Home Page



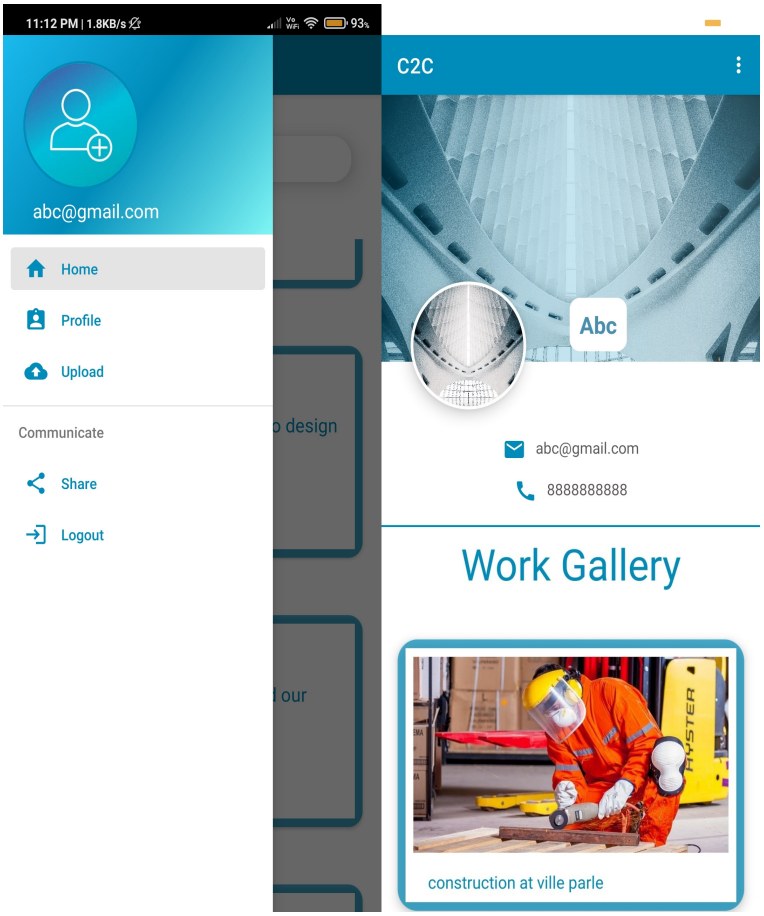
4.6.1: Home Page View

4.7 Contractor Set Up Profile



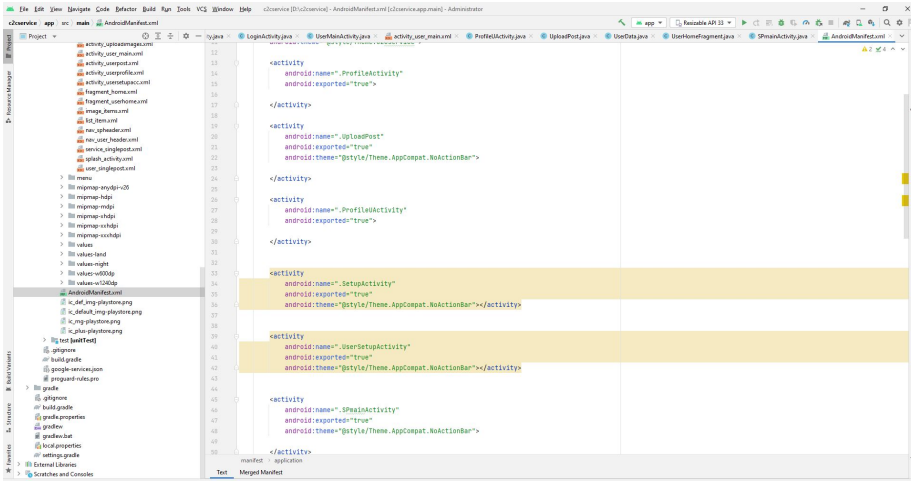
4.7.1: Setup Profile

4.8 Contractors Profile Page

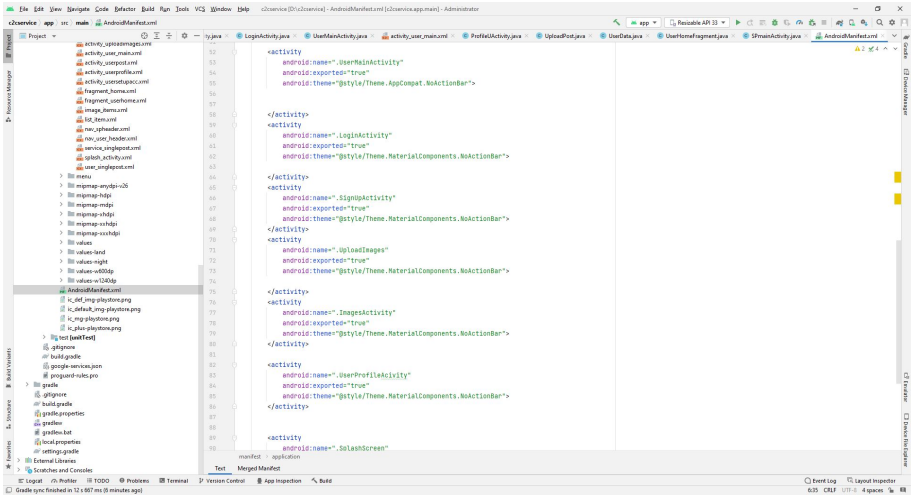


4.8.1: Profile and Posts

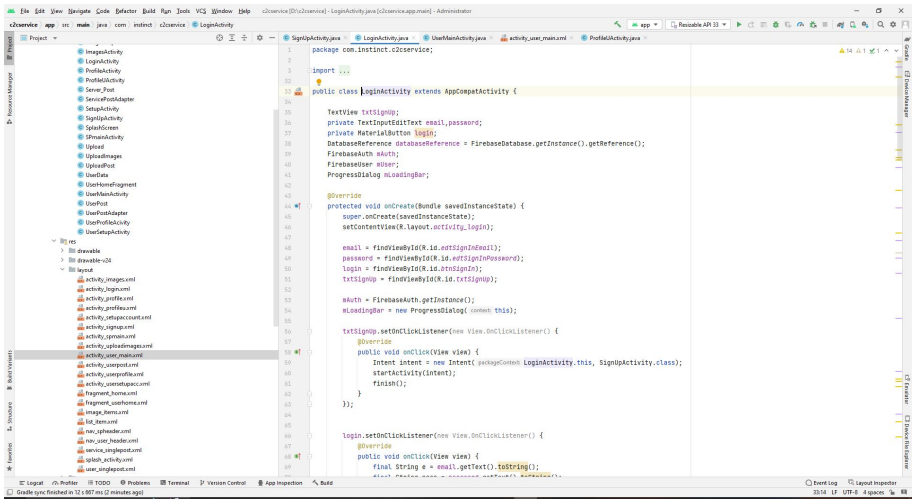
4.9 Code 1



4.10 Code 2

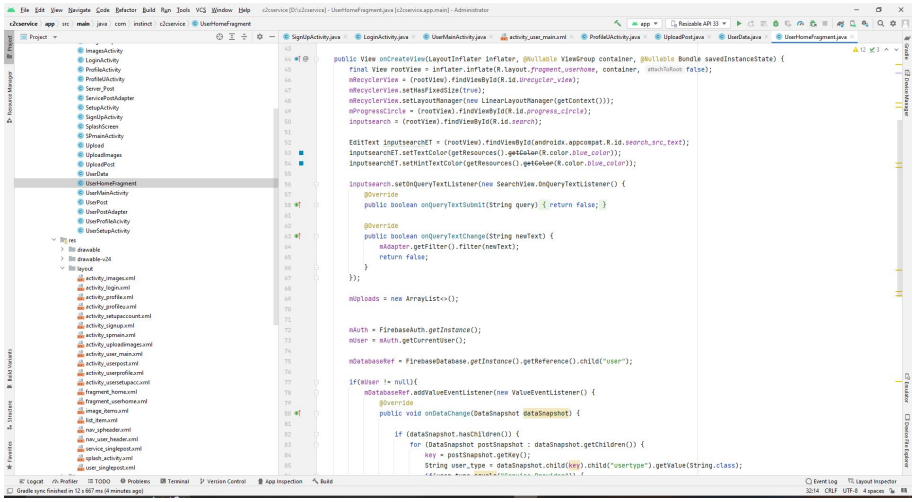


4.11 Code 3



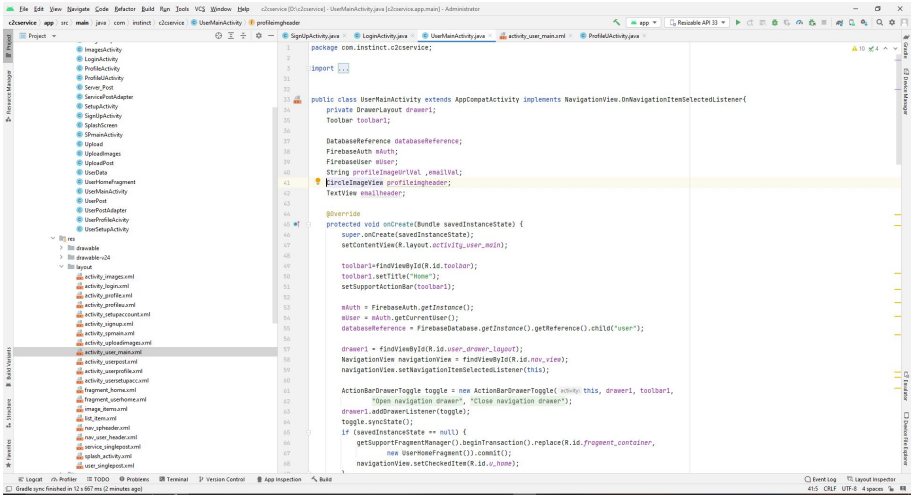
```
package com.instinct.c2cservice;

import ...

public class LoginActivity extends AppCompatActivity {

    TextView txtSignUp;
    private TextInputEditText email,password;
    private MaterialButton login;
    DatabaseReference databaseReference = FirebaseDatabase.getInstance().getReference();
    FirebaseAuth mAuth;
    FirebaseUser mUser;
    ProgressDialog mLoadingBar;

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        email = findViewById(R.id.edtSignInEmail);
        password = findViewById(R.id.edtSignInPassword);
        login = findViewById(R.id.btnSignIn);
        txtSignUp = findViewById(R.id.txtSignUp);

        mAuth = FirebaseAuth.getInstance();
        mLoadingBar = new ProgressDialog( context: this);

        txtSignUp.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent = new Intent( packageContext: LoginActivity.this, SignUpActivity.class);
                startActivity(intent);
                finish();
            }
        });


        login.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                final String e = email.getText().toString();
```

4.12 Code 4



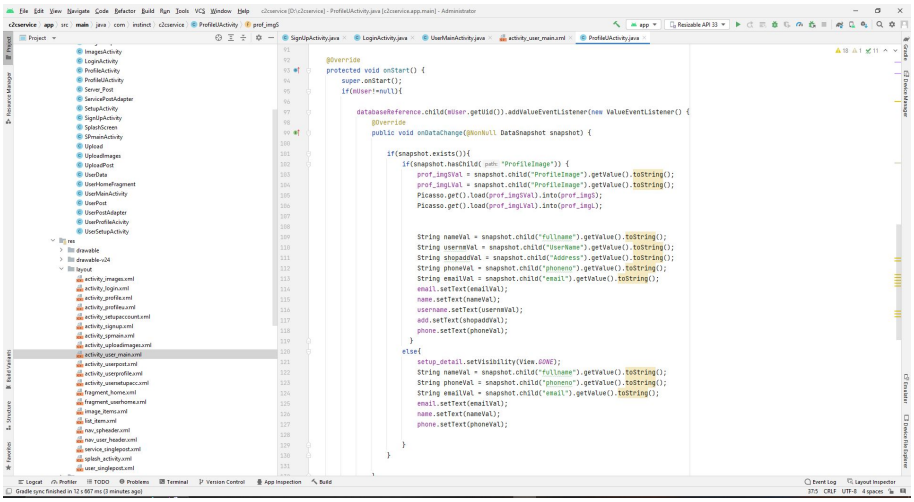
```
    public View onCreateView(LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
        final View rootView = inflater.inflate(R.layout.fragment_userhome, container,  attachToRoot: false);
        mRecyclerView = (rootView).findViewById(R.id.recycler_view);
        mRecyclerView.setHasFixedSize(true);
        mRecyclerView.setLayoutManager(new LinearLayoutManager(getContext()));
        mProgressCircle = (rootView).findViewById(R.id.progress_circle);
        inputsearch = (rootView).findViewById(R.id.search);

        EditText inputsearchET = (rootView).findViewById(androidx.appcompat.R.id.search_src_text);
        inputsearchET.setTextColor(getResources().getColor(R.color.blue_color));
        inputsearchET.setHintTextColor(getResources().getColor(R.color.blue_color));

        inputsearch.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
            @Override
            public boolean onQueryTextSubmit(String query) { return false; }

            @Override
            public boolean onQueryTextChange(String newText) {
                mAdapter.getFilter().filter(newText);
                return false;
            }
        });

        mUploads = new ArrayList<>();


        mAuth = FirebaseAuth.getInstance();
        mUser = mAuth.getCurrentUser();

        mDatabaseRef = FirebaseDatabase.getInstance().getReference().child("user");

        if(mUser != null){
            mDatabaseRef.addValueEventListener(new ValueEventListener() {
                @Override
                public void onDataChange(DataSnapshot dataSnapshot) {

                    if (dataSnapshot.hasChildren()) {
                        for (DataSnapshot postSnapshot : dataSnapshot.getChildren()) {
                            key = postSnapshot.getKey();
                            String user_type = dataSnapshot.child(key).child("usertype").getValue(String.class);
```

4.13 Code 5



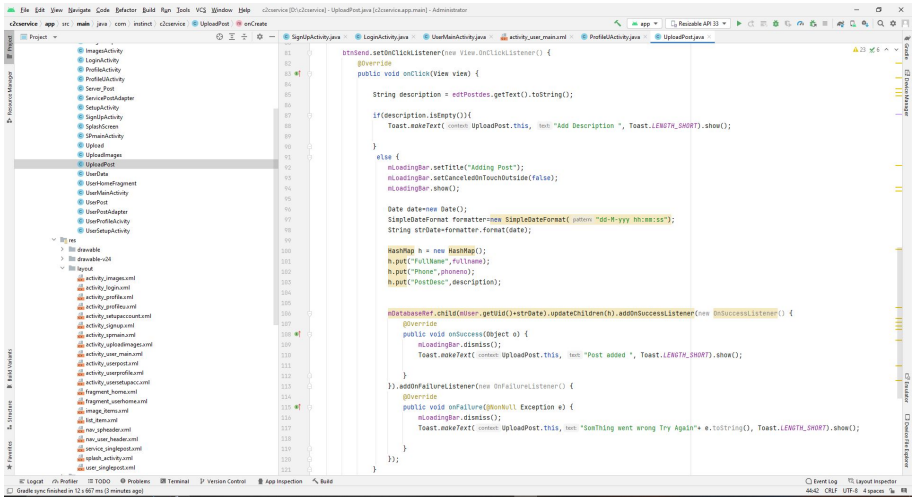
```
package com.instinct.c2cservice;

import ...

public class UserMainActivity extends AppCompatActivity implements NavigationView.OnNavigationItemSelectedListener{
    private DrawerLayout drawer1;
    Toolbar toolbar1;

    DatabaseReference databaseReference;
    FirebaseAuth mAuth;
    FirebaseUser mUser;
    String profileImageUrVal ,emailVal;
    CircleImageView profileimgheader;
    TextView emailheader;

    @Override
    protected void onCreate(@Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_user_main);

        toolbar1=findViewById(R.id.toolbar);
        toolbar1.setTitle("Home");
        setSupportActionBar(toolbar1);

        mAuth = FirebaseAuth.getInstance();
        mUser = mAuth.getCurrentUser();
        databaseReference = FirebaseDatabase.getInstance().getReference().child("user");

        drawer1 = findViewById(R.id.user_drawer_layout);
        NavigationView navigationView = findViewById(R.id.nav_view);
        navigationView.setNavigationItemSelectedListener(this);

        ActionBarDrawerToggle toggle = new ActionBarDrawerToggle( activity: this, drawer1, toolbar1,
                "Open navigation drawer", "Close navigation drawer");
        drawer1.addDrawerListener(toggle);
        toggle.syncState();
        if (savedInstanceState == null) {
            getSupportFragmentManager().beginTransaction().replace(R.id.fragment_container,
                    new UserHomeFragment()).commit();
            navigationView.setCheckedItem(R.id.u_home);
        }
```

4.14 Code 6



```
    @Override
    protected void onStart() {
        super.onStart();
        if(mUser!=null){

            databaseReference.child(mUser.getUid()).addValueEventListener(new ValueEventListener() {
                @Override
                public void onDataChange(@NonNull DataSnapshot snapshot) {

                    if(snapshot.exists()){
                        if(snapshot.hasChild( path: "ProfileImage")) {
                            prof_imgSVal = snapshot.child("ProfileImage").getValue().toString();
                            prof_imgLVal = snapshot.child("ProfileImage").getValue().toString();
                            Picasso.get().load(prof_imgSVal).into(prof_img5);
                            Picasso.get().load(prof_imgLVal).into(prof_imgL);


                            String nameVal = snapshot.child("fullname").getValue().toString();
                            String usernameVal = snapshot.child("UserName").getValue().toString();
                            String shopaddVal = snapshot.child("Address").getValue().toString();
                            String phoneVal = snapshot.child("phoneno").getValue().toString();
                            String emailVal = snapshot.child("email").getValue().toString();
                            email.setText(emailVal);
                            name.setText(nameVal);
                            username.setText(usernameVal);
                            add.setText(shopaddVal);
                            phone.setText(phoneVal);
                        }
                        else{
                            setup_detail.setVisibility(View.GONE);
                            String nameVal = snapshot.child("fullname").getValue().toString();
                            String phoneVal = snapshot.child("phoneno").getValue().toString();
                            String emailVal = snapshot.child("email").getValue().toString();
                            email.setText(emailVal);
                            name.setText(nameVal);
                            phone.setText(phoneVal);

                        }
                    }
```

4.15 Code 7


btnSend.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View view) {
String description = edtPostdes.getText().toString();
if(description.isEmpty()){
Toast.makeText(context: UploadPost.this, text: "Add Description ", Toast.LENGTH_SHORT).show();
}
else {
mLoadingBar.setTitle("Adding Post");
mLoadingBar.setCanceledOnTouchOutside(false);
mLoadingBar.show();
Date date=new Date();
SimpleDateFormat formatter=new SimpleDateFormat(pattern: "dd-M-yyy hh:mm:ss");
String strDate=formatter.format(date);
HashMap h = new HashMap();
h.put("FullName",fullname);
h.put("Phone",phoneno);
h.put("PostDesc",description);
mDatabaseRef.child(mUser.getUid()+strDate).updateChildren(h).addOnSuccessListener(new OnSuccessListener() {
@Override
public void onSuccess(Object o) {
mLoadingBar.dismiss();
Toast.makeText(context: UploadPost.this, text: "Post added ", Toast.LENGTH_SHORT).show();
}
}).addOnFailureListener(new OnFailureListener() {
@Override
public void onFailure(@NonNull Exception e) {
mLoadingBar.dismiss();
Toast.makeText(context: UploadPost.this, text: "SomThing went wrong Try Again"+ e.toString(), Toast.LENGTH_SHORT).show();
}
});
}

5 Test Cases

Table 6.1: Test Case - Login and Register

Test Case ID	Test Case Name	Test Data	Expected Output	Actual Output	Result
1	User enter user id and password	Enters the correct user id and password	Log in Successful	Home Page	Pass
2	User enter user id and password	New user Enters the user id and password	Not Registered	Prompt error Showing Check Your Credentials	Pass
3	New User Registration	New user provides Required details	Registered Successfully	Prompt Message Registered Successfully	Pass
4	New User Registration	New user doesn't provide Required details	Not Registered	Prompt error Showing Please fill all details	Pass
5	User enter user id and password	Valid user id and password which doesn't exist in Database	Prompt error	Login Page	Pass
6	User enter user id and password	Invalid user id and password which contains in Database	Prompt error	Prompt error	Pass

Table 6.2: Test Case - Homepage And Profile

Test Case ID	Test Case Name	Test Data	Expected Output	Actual Output	Result
1	User Homepage	Side-menu And Feed	Open Side-menu Scroll through Posts	Scrolling Page and Open Side-Menu when click	Pass
2	User Profile	Setup Profile	Profile Updated successful	Profile Updated successful	Pass

Table 6.2: Test Case - Add Posts

Test Case ID	Test Case Name	Test Data	Expected Output	Actual Output	Result
1	User creates post	Post Description	Prompt Message showing Post Added	Prompt Message showing Post Added	Pass
2	User Visits Different Profile	Profile details and Work Gallery	Profile Opens	Profile Showing all details	Pass

Table 6.2: Test Case - Open App And Log Out

Test Case ID	Test Case Name	Test Data	Expected Output	Actual Output	Result
1	User Logged Out	press log out button	Prompt Message showing Logged Out	Prompt Message showing Logged Out	Pass
2	User Opens app after login	Logged In User	Should Already be logged In	User Already logged In	Pass

6 Limitations

- It needs internet to be accessed.
- It does not have a feature to connect where users can only view the service providers as per his/her needs.
- User cannot post pictures if needed.

7 Future Enhancements

- For User, Option to choose the type of service provider.
- For Contractors, the project they are working on.
- Feature of having communication via message for user and contractor.
- Rating system for users to rate the service provider.

8 User Manual

Part 1 – Register
Upon opening the application, the user will be greeted with the registration screen. If the user has no account, the user can click on register and register self.
After verification of the user id, the User's account will be saved in our database. The user can now proceed to log in.

Part 2 – Login
The user needs to enter the email first and then the password. If there is an active internet connection, the user can proceed to log in.

Part 3 –Setup Profile
Users can access/add/update personal details (eg. Name, Contact No.). Users can update these details anytime.

Part 4 – Feed
Users can access feeds and posts added by service providers.

Part 5 – Posts
Users can add and view service providers and can add their posts and requirements for contractors to know.

Part 6 – Contact
Users can view service providers profile and can contact them as per their choice of work.

9 Bibliography

9.1 Web References

[1.] http://developer.android.com/docs/
[2.] https://firebase.google.com/docs
[3.] https://www.youtube.com/user/Firebase
[4.] https://stackoverflow.com/
[5.] https://www.lucidchart.com/
[6.] https://www.geeksforgeeks.org/unified-modeling-language-uml-introduction/
[7.] https://www.edrawmax.com/online/
[8.] https://youtube.com/playlist?list=PLlGT4GXi8_8dDK5Y3KCxuKAPpil9V49rN