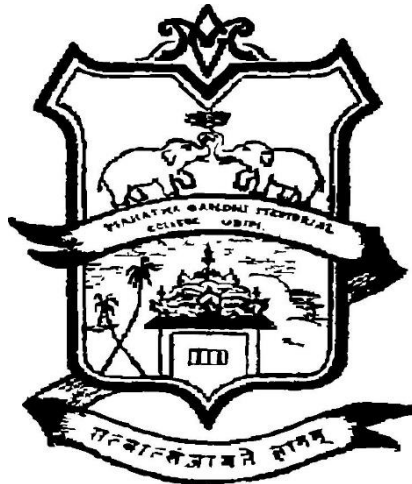


MAHATMA GANDHI MEMORIAL COLLEGE
UDUPI – 576102
Accredited by NAAC with “A+” Grade (CGPA 3.36)



PROJECT REPORT 2022-23

“E-BANKING APPLICATION”

DEVELOPED BY

Mr. Goutham
Mr. Anmol S Shetty
Mr. Prajwal

Reg.No:201161522116
Reg.No:201161522109
Reg.No:201161522133

Under the guidance of

“Prof. Mithun S”
Dept. of Computer Science

Submitted to the Mangalore University
in partial fulfilment of the Award of
Bachelor of Computer Application

MANGALORE UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE
MAHATMA GANDHI MEMORIAL COLLEGE

UDUPI-576102

ACKNOWLEDGEMENT

Behind every achievement, there is a sea of gratitude to those who have activated this project.

The magnitude of this project demanded the co-operation, guidance and help of many people.

We have been fortunate enough to have this in the entire task of completion of our project on “**E-Banking Application**”

We would like to thank our principal **Prof. Lakshmi Narayan Karanth** for giving us opportunity to carry out our project.

We thank **Dr. M. Vishwanath Pai**, a source of inspiration and encouragement, head of the Department of Computer Science, Mahatma Gandhi Memorial College, Udupi for having permitted us to carry out our project work.

We are extremely grateful to express our overwhelming gratitude to our guide **Prof. Mithun S** Lecturer of Computer Science Department, Mahatma Gandhi Memorial College Udupi for giving us valuable guidance to undertake this project.

Last but not the least, we are indebted to all teaching and non-teaching staff members, Mahatma Gandhi Memorial College Udupi for making this project successful.

Thanking you

**Goutham
Anmol S Shetty
Prajwal**



GAP

INDEX

1. Introduction

- i. Introduction of the System
 - a. Project Title
 - b. Category
 - c. Overview
- ii. Objectives of the System
- iii. Scope of the System
- iv. End Users
- v. Software/Hardware used for the development
- vi. Software/Hardware required for the implementation

2. SRS

- i. Introduction
- ii. Overall Description
 - a. Product perspective
 - b. Product Functions
 - c. User Characteristics
 - d. General Constraints
 - e. Assumptions
- iii. Functional Requirements
 - a. Modules
 - b. Module Description

3. System Design

- i. Introduction
- ii. Description of Programs
 - a. Context Flow Diagram
 - b. Data Flow Diagram
 - c. Data dictionary

4. Database Design

- i. Introduction
- ii. Database Diagrams

5. Program Code Listing

6. User Interfaces

7. Testing

- i. Introduction
- ii. Test Reports
 - a. Unit Testing
 - b. Integration Testing
 - c. System Testing
- iii. Test Plan
- iv. Test Cases

8. Limitations

9. Scope For Enhancements

10. Abbreviation and Acronyms

11. Bibliography/References

1.Introduction

i. Introduction to the System:

GAP is a simple and user-friendly E-banking app that aims in easifying the banking operations. With its minimalist design and straightforward interface, user can completely rely on GAP. It allows customers to conveniently access and manage their finances from anywhere, at any time, using android smartphones. This application provide features like account balance inquiries, fund transfers, and more, offering a secure and efficient way for individuals and businesses to conduct banking activities remotely

a. Project Title:

GAP

b. Category:

Android Application

c. Overview:

GAP is an Android project that allows users to perform banking operations without physically visiting to the bank. Users can transfer fund directly by scanning QR code or adding beneficiary account. This application also facilitates uploading user profiles that is visible to all the connected user. This app provides the utilities such as calculator, ATM search, chatbot and video guide. Using which user can estimate the EMI amount and rate of interest on their loans and rate of returns on their investments.

ii. Objectives of the System:

- To provide a reliable platform to perform banking operations.
- To allow the users to transfer the money safely and securely over the internet.
- It aims to provide customers with convenient access to banking services anytime and anywhere
- It aims to make banking services accessible to a broader customer base, including those in remote areas or with limited mobility
- It aims to reduce operational costs for banks by replacing traditional paper-based processes with electronic systems
- It aims to provide faster and more efficient banking services. It allows for real-time transactions, instant fund transfers, and quicker access to account information, minimizing processing time and enhancing customer experience

iii. Scope of the System:

GAP allows the users to perform various financial transactions, including fund transfers, account balance inquiries, and transaction history. With the increasing use of smartphones and tablets, e-banking application have a significant scope in mobile banking. Users can easily download and install it in their smartphone.

iv. End Users:

End users are the users who holds an account in the Teachers' bank and registered themselves with GAP either with their debit card details or registered phone number. After successful registration these users can perform banking operations.

v. Software/Hardware used for the development:

- **Software:**

- **Front End:**

- **Android studio:** Android Studio is the official Integrated Development Environment (IDE) for Google's Android operating system, built on JetBrains IntelliJ IDEA software and designed specifically for android development. Android Studio provides more features that enhance our productivity while building Android apps. It has a flexible Gradle-based build system and it also has a fast and feature-rich emulator for app testing. Android Virtual Device (Emulator) is used to run and debug apps in the Android Studio. Android Studio contains all the Android tools to design, test, debug, and profile the application. The Android Studio uses Gradle to manage the project, a Build Automation Tool. It provides a unified environment where we can build apps for Android phones, tablets, etc. The official language for Android Development is Java. The app module contains the following folders:
 - **manifests:** It contains the AndroidManifest.xml file. Most of the code is generated automatically in this file, we need not change anything. The AndroidManifest.xml file contains information about the package, including components of the application such as activities, services, broadcast receivers, content providers, etc.
 - **java:** It contains the source code of java files including the JUnit test code.
 - **res:** It contains all non-code resources, UI strings, XML Layouts, and bitmap images
 - **Figma:** Figma is a web-based collaborative design tool used for creating user interfaces, prototypes, and design systems. It offers a range of features and capabilities that make it popular among designers and design teams.

- **Back End:**

- **Firebase:** It is a Google backed application development software that enables the developers to develop android apps. The advantage of using online databases is that there are less chances of data being lost. The insertion and retrieval of data makes use of Firebase. Firebase is a Backend-as-a-Service, and it is a real time database which is basically designed for mobile applications.
 - **Authentication:** It aims to make building secure authentication systems easy, while improving the sign-in

and onboarding experience for end users. It provides an end-to-end solution supporting email and password accounts, phone auth and Google, Twitter login and more. Firebase Authentication provides backend services, easy-to-use SDKs, and readymade UI libraries to authenticate users to the app. Firebase Authentication uses Firebase Dynamic Links to send the email link to the mobile devices.

- **Realtime Database:** Firebase Realtime Database is a cloud-hosted database in which data is stored as JSON. The data is synchronised in real-time to every connected client. It is a NoSQL database from which we can store and sync the data between our users in real-time. It uses data synchronisation instead of using HTTP requests, due to this any connected device receives the updates within milliseconds. It is capable of providing all offline and online services. It ships with mobile SDKs, which allow us to build our app without the need for servers. Real-time syncing makes it easy for our users to access their data from any device, be it a web or mobile.
- **Firebase Storage:** Firebase Storage is a cloud storage solution provided by Google as part of the Firebase suite of services. It enables developers to store and serve user-generated content, such as images, videos, and other files, for their web and mobile applications.

- **Hardware:**

CPU: 32bit/64bit

RAM: Minimum 8GB or more

Hard Disk: Minimum 100Gb or more

vi. Software/Hardware used for the implementation:

Software:

Our Android project, GAP, uses common software tools like the Android operating system and Android Studio. Hence on the software side, we expect the user to use an Android Operating System based smartphone with minimum of Android Version 5.0 (Lollipop).

Hardware:

To run our Android app smoothly, use any Android-compatible smartphone or tablet. Make sure it has enough processing power, memory, and storage space. It should also support Wi-Fi, GPS, mobile data and optional requirements like fingerprint sensor, face recognition sensor for seamless connectivity.

RAM: Minimum 2GB

Storage: Minimum 8GB

2. Software Requirement Specification (SRS)

i. Introduction:

A Software Requirement Specification (SRS) is the complete description about what the software will do and how the software is expected to perform it. It also describes the functionality that the software needs to fulfil all the user needs. It is a formal report that enables the users to review whether SRS is according to their requirements.

ii. Overall Description:

a. Product Perspective:

From a product perspective, GAP is the electronic delivery of various banking services and products through android device. It allows customers to access and manage their financial accounts, perform transactions, and avail banking services remotely, without the need for physical visits to a bank branch

b. Product Functions:

- It enables people to experience the product starting from a simple registration process.
- It functions in providing a seamless user-interface for simple banking activities.

c. User Characteristics:

- GAP users rely on smartphones as their primary device for financial transactions. They are comfortable using mobile applications and appreciate the convenience of using their phones for various tasks, including making payments.
- The users understand the importance of security in digital transactions. They trust the security measures implemented by GAP, such as t-pin authentication and biometric authentication, to protect their payment information.

d. General Constraints:

App should run on any android device where the android version is more than 5.

e. Assumptions:

- GAP assumes that user already have registered for e-banking and they should hold an account in the bank.
- The project will be updated in the near future with new features like currency converter, bill payments, recharge, chatting, enhancing card maintenance etc.

iii. Functional requirements:

a. Modules:

1. User Registration
2. User Login
3. Calculator

4. ATM search
5. Help and Support
6. Fund Transfer
7. User Profile

b. Module Description:

1. User Registration:

This module describes the registration functionalities for a user of this application. A new user can register either using their debit card or by registered phone number.

2. User Login:

Once the user is registered successfully, he/she can login to the app either using login pin or biometric authentication. User can reset the login pin by using email verification.

3. Calculator:

This module helps the user to calculate the EMI amount and total interest payable on the loan they are planning to take. It also calculates the rate of returns on their deposits.

4. ATM search:

User can find all nearby ATMs with navigation to the nearest ATM.

5. Help and Support:

GAP provides utility like chatbot and video guide to assist the user to use the app.

6. Fund Transfer:

User can quickly and securely transfer the fund by directly scanning the QR code or by adding beneficiary account.

7. User Profile:

It provides the details of current users as provided in the bank. User can also edit their profile pick that will be shown to all the connected users. User can also de-register from the app.

3. System Design

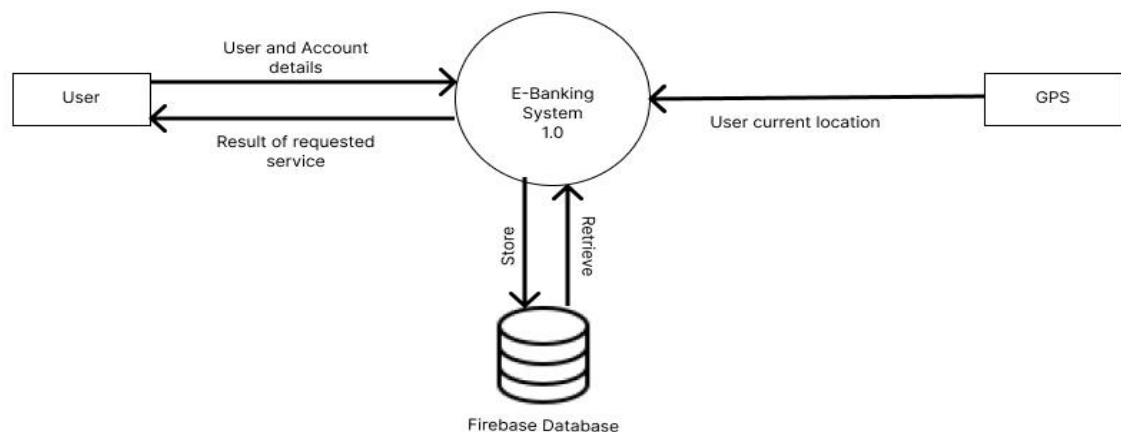
i. Introduction:

System design is the process of defining elements of a system like modules, architecture, components and their interfaces and data for a system based on the specified requirements. It is the process of defining, developing and designing systems which satisfies the specific needs and requirements of a business or organisation.

ii. Description of Programs:

a. Context flow Diagram:

CFD is one in which the entire system is treated as a single process and all its inputs, outputs, sinks and sources are identified. A context diagram, sometimes called a level 0 data-flow diagram, is drawn in order to define and clarify the boundaries of the software system. It identifies the flows of the information between the system and external entities. The entire software system is shown as a single process. It is used to establish the context and boundaries of the system to be modelled and identify the relationship of the system with the external entities.



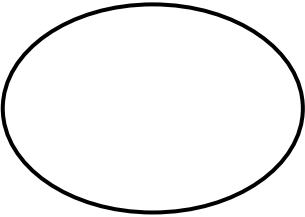



b. Data Flow Diagram:

A data-flow diagram is a way of representing a flow through a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data flow diagram has no control flow; there are no decision rules and no loops. Levels in DFD are numbered 0, 1, 2 and beyond.

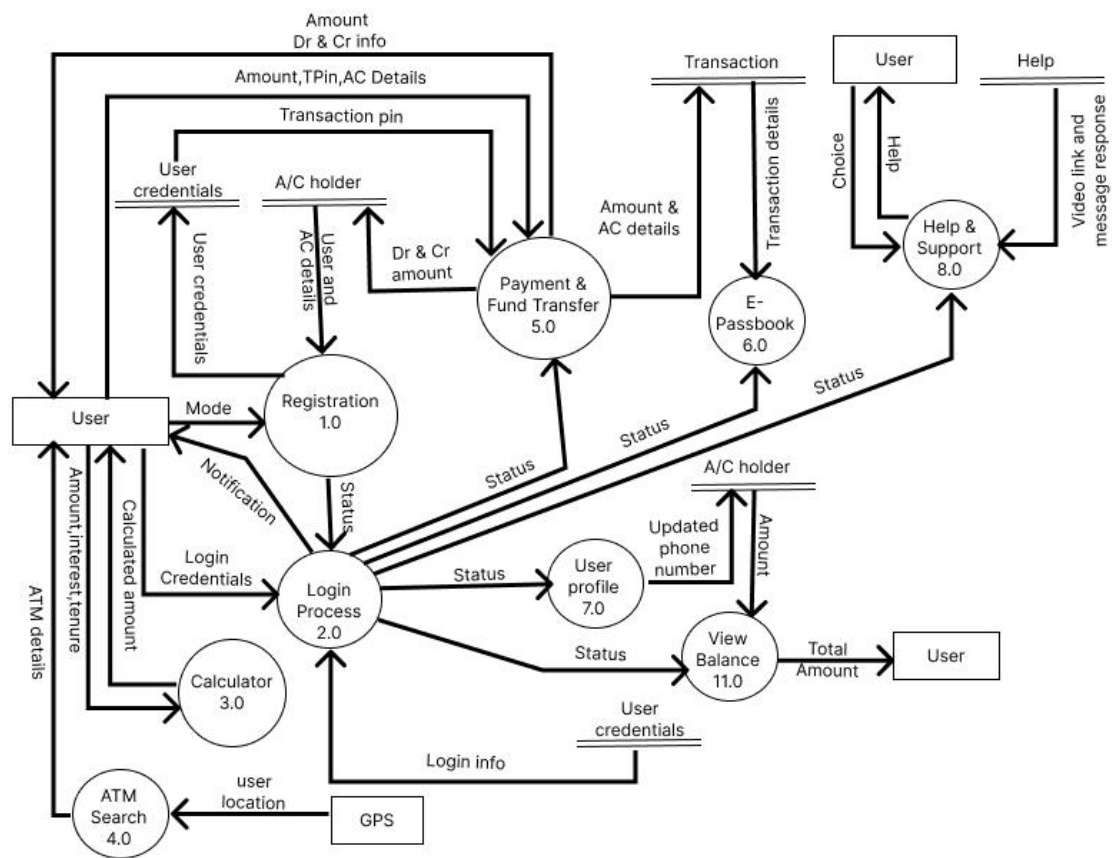
The DFD server two purposes:

- To provide an indication of flow of data are transformed as they move through the system.
- To depict the function that transforms the data flow.

Notations for DFD Diagram:

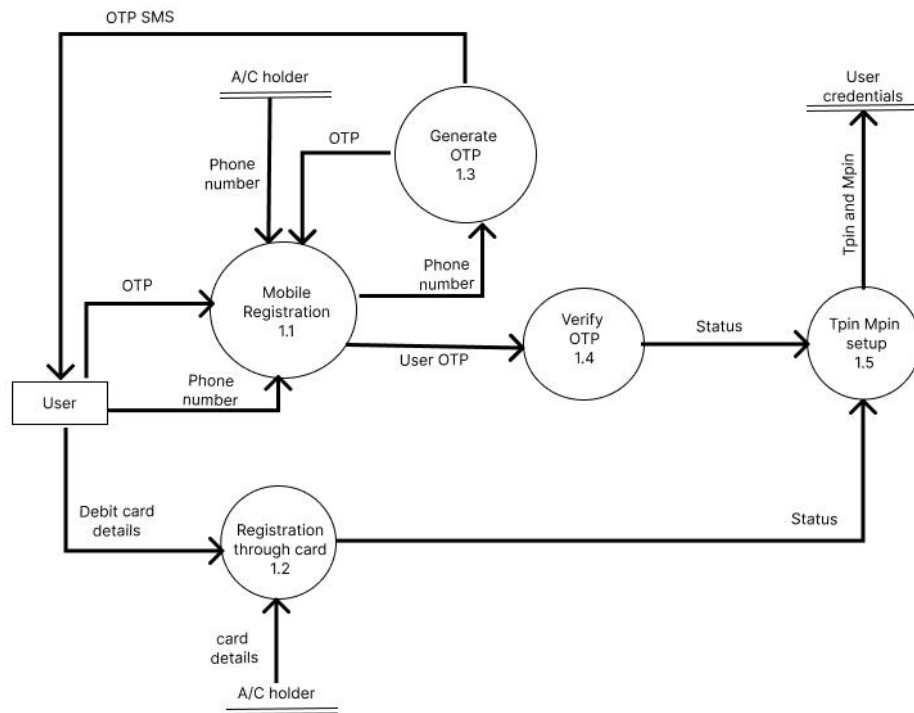
Notations	Description
	A circle represents a Process that performs some transformation of input data to yield output data.
	A rectangle is used to represent an external entity i.e. a source and a sink. System that produces information by the software is a Source and a system that receives or stores information produced by the system is a Sink .
	The open box represents a repository of data stored that is used by the software.
	An arrow represents one or more data items or data objects and is used to connect processes to each other or to sources and sinks. The arrow head indicates direction of data flow.

Level 1 DFD

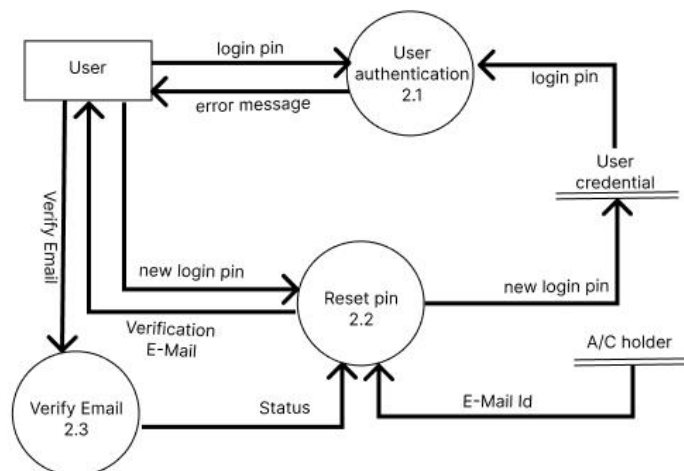


Level 2 DFD

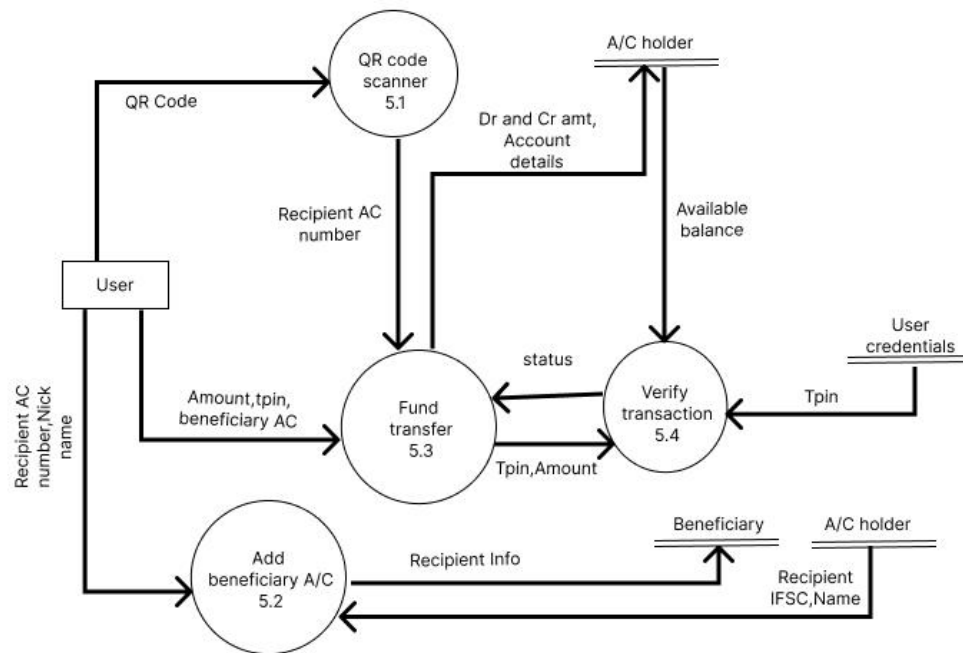
Registration 1.0



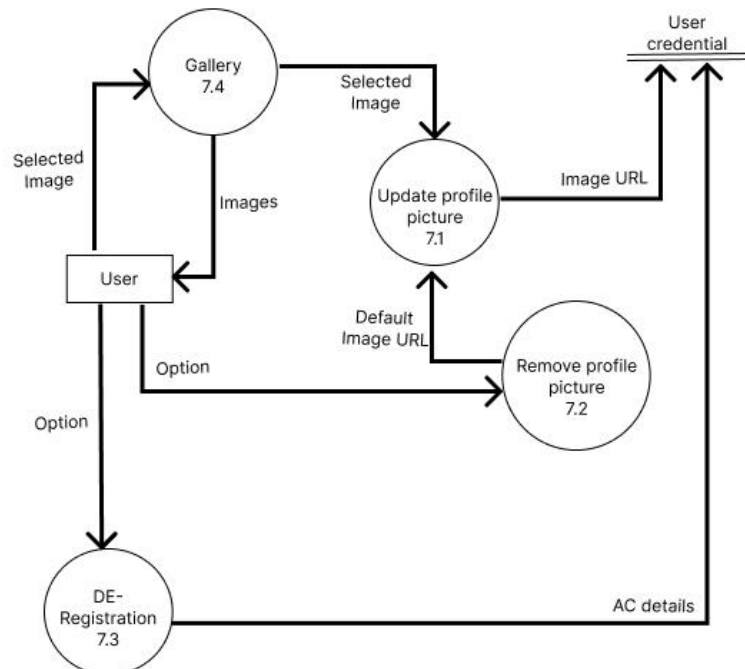
Login Process 2.0



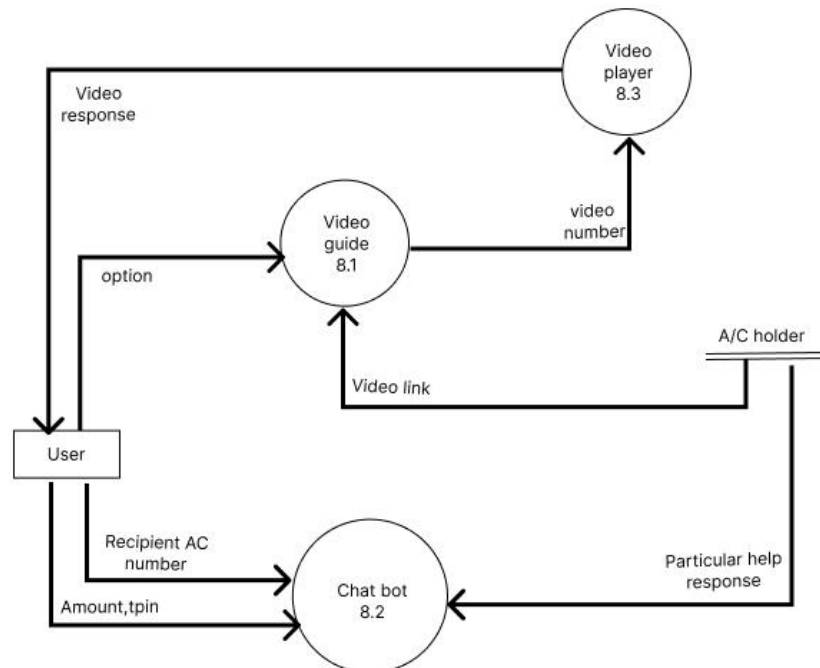
Payment and Fund Transfer 3.0



User Profile 4.0



Help and Support 8.0



c. Data Dictionary

Login and Transaction Pin : [0-9]{4}

Phone number : ^[6,7,8,9]+[0-9]{9}\$

Card Number : [0-9]{16}

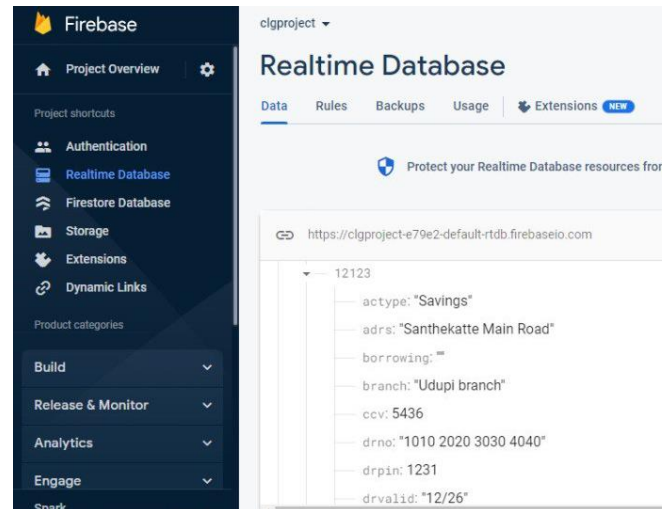
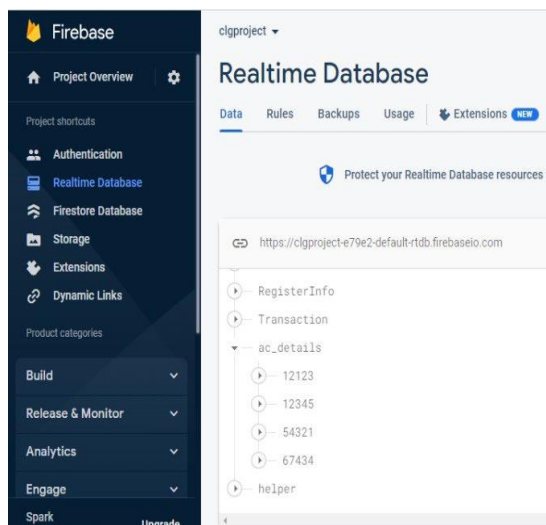
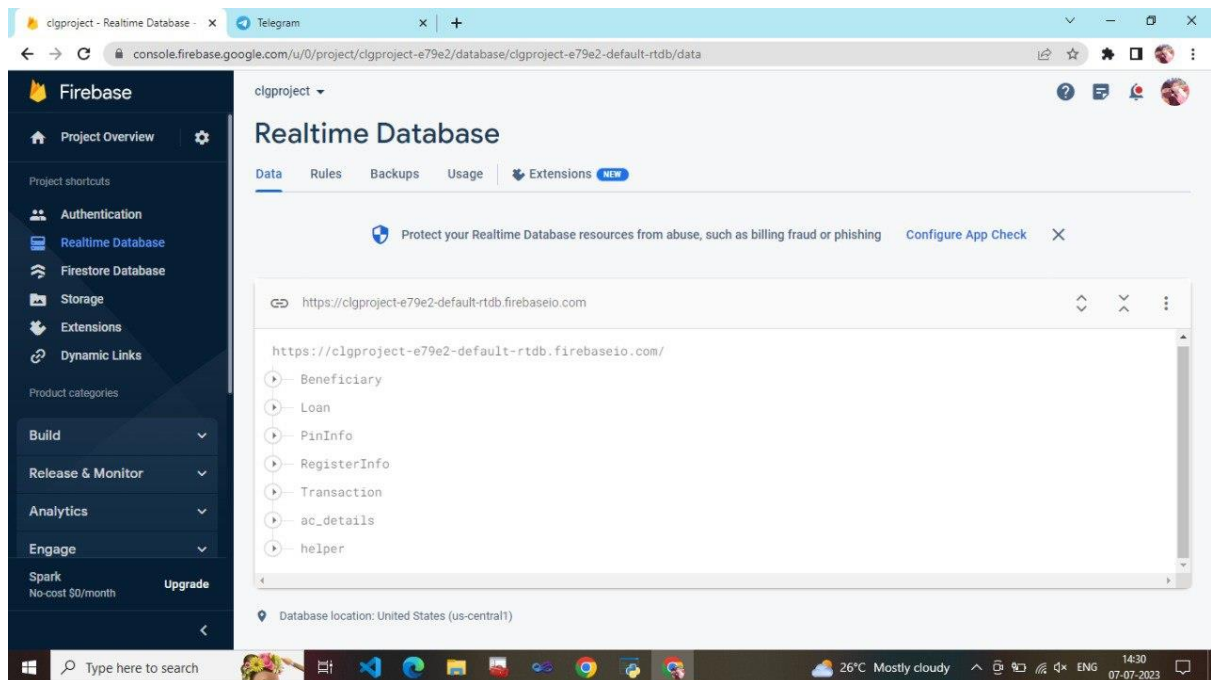
4. Database Design

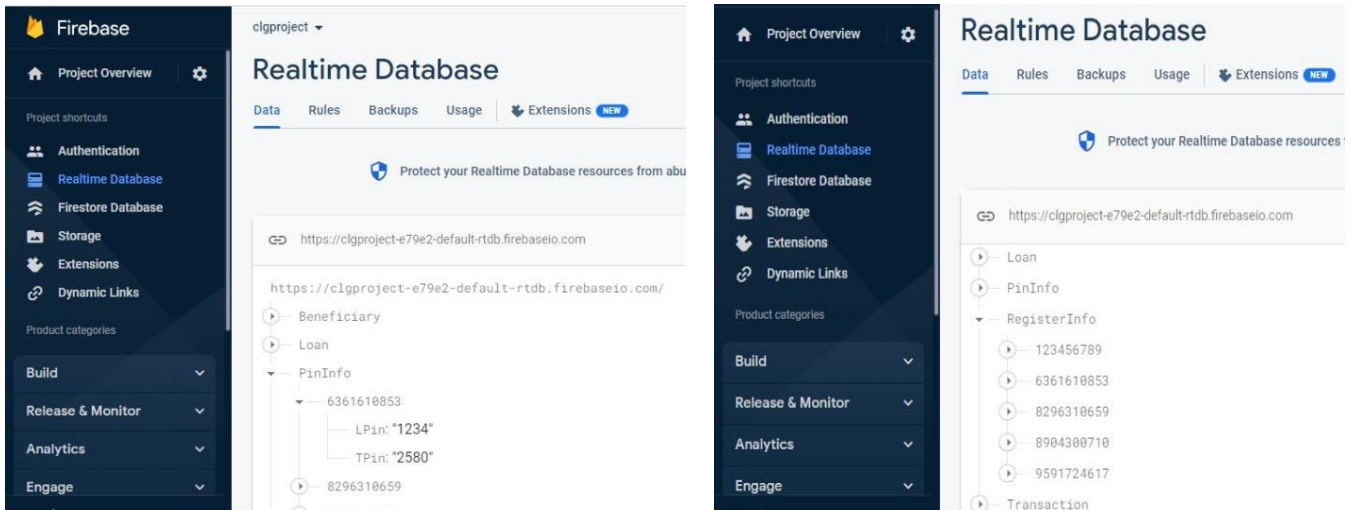
i. Introduction:

Database design is the organisation of data according to database model. The designer determines what data must be stored and how the data elements interrelate. With this information, they can begin to fit the data to the database model. Database management system manages the data accordingly. Database design involves classifying data and identifying interrelationships.

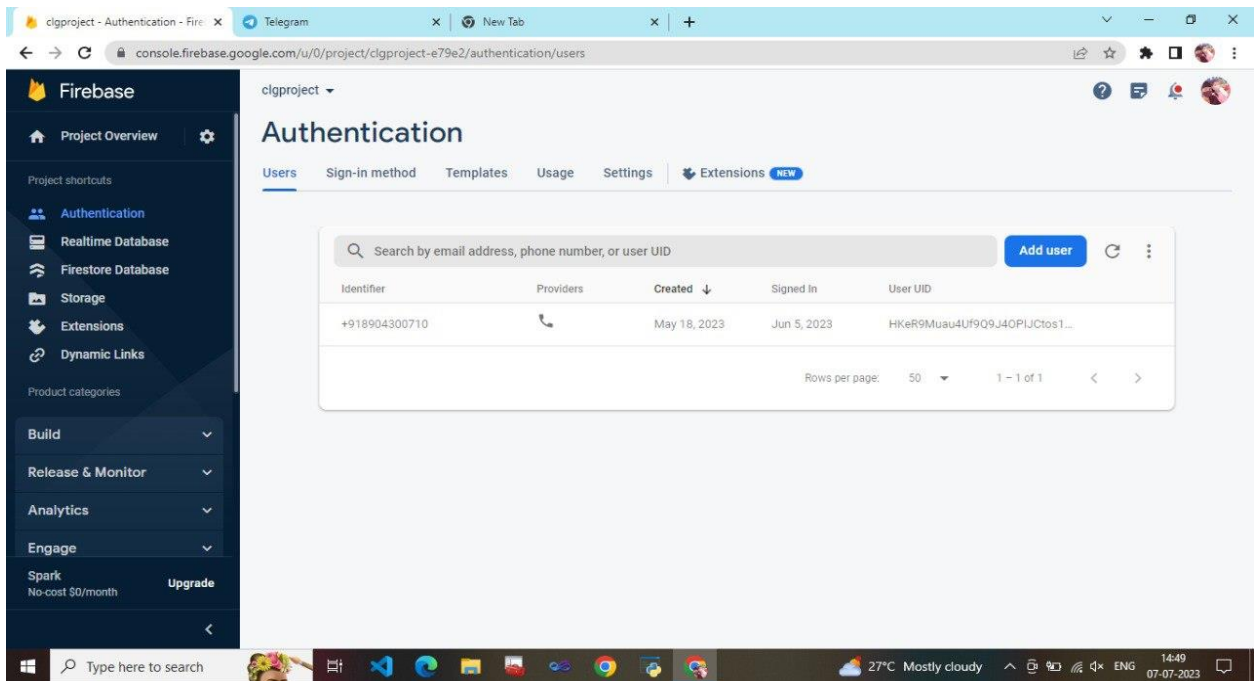
ii. Database Diagrams:

a. Realtime Database:





b. Authentication:



c. Storage:

The screenshot shows the Firebase Storage console for a project named 'clgproject'. The left sidebar contains the 'Project Overview' and 'Project shortcuts' sections. The main area displays the 'Storage' section with tabs for 'Files', 'Rules', 'Usage', and 'Extensions'. A warning banner at the top states: 'Protect your Storage resources from abuse, such as billing fraud or phishing'. Below this, a table lists the contents of the storage bucket 'gs://clgproject-e79e2.appspot.com'. The table has columns for 'Name', 'Size', 'Type', and 'Last modified'. The contents are folders: 'Anmol-6361610853/', 'Divya-9591724617/', 'Goutham-8904300710/', 'Prajwal-8296310659/', and 'signature/'.

Name	Size	Type	Last modified
Anmol-6361610853/	—	Folder	—
Divya-9591724617/	—	Folder	—
Goutham-8904300710/	—	Folder	—
Prajwal-8296310659/	—	Folder	—
signature/	—	Folder	—

This screenshot shows the 'signature' directory within the Firebase Storage console. The breadcrumb path is 'gs://clgproject-e79e2.appspot.com > signature'. The table lists five image files:

Name	Size	Type	Last modified
signanmol.jpeg	12.81 KB	image/jpeg	Jul 6, 2023
signdivya.jpeg	14.13 KB	image/jpeg	Jul 6, 2023
signgoutham.jpeg	14.53 KB	image/jpeg	Jul 6, 2023
signprajwal.jpeg	14.82 KB	image/jpeg	Jul 6, 2023

This screenshot shows the details of the 'signanmol.jpeg' file. The breadcrumb path is 'gs://clgproject-e79e2.appspot.com > signature'. The file details are as follows:

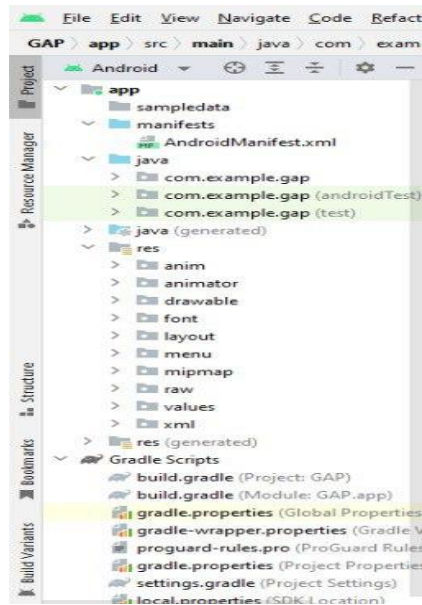
Name	Size	Type	Last modified
signanmol.jpeg	12.81 KB	image/jpeg	Jul 6, 2023
signdivya.jpeg	14.13 KB	image/jpeg	Jul 6, 2023
signgoutham.jpeg	14.53 KB	image/jpeg	Jul 6, 2023
signprajwal.jpeg	14.82 KB	image/jpeg	Jul 6, 2023

File details for 'signanmol.jpeg':

- Name: signanmol.jpeg
- Size: 13,116 bytes
- Type: image/jpeg
- Created: Jul 6, 2023, 9:05:18 PM
- Updated: Jul 6, 2023, 9:05:18 PM

5. Program Code Listing

Project Structure:



Registration Activity:

```
package com.example.gap;

import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;

import android.content.Intent;

import android.content.SharedPreferences;

import android.content.res.ColorStateList;

import android.graphics.drawable.RippleDrawable;

import android.os.Build;

import android.os.Bundle;

import android.view.MotionEvent;

import android.view.View;

import android.view.WindowManager;

import android.widget.LinearLayout;

import android.widget.RadioButton;
```

```

import android.widget.RadioGroup;

import android.widget.Toast;


public class Registration extends AppCompatActivity {

    SharedPreferences u_details;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_registration);

    }


    public void onClick(View v)

    {

        RadioGroup rgrp=findViewById(R.id.radioGroup);

        Intent i=null;

        if(rgrp.getCheckedRadioButtonId()==R.id.radioButtonDebit)

            i=new Intent(this,RegistrationCard.class);

        else if(rgrp.getCheckedRadioButtonId()==R.id.radioButtonMbl)

            i=new Intent(this,RegisterMbl.class);

        else

            Toast.makeText(this, "Please select the mode of registration",
Toast.LENGTH_SHORT).show();

        if(i!=null) {

            startActivity(i);

        }

    }

}

```

```

@Override

protected void onRestart() {

    u_details=getSharedPreferences("user_details",MODE_PRIVATE);

    if(u_details.getBoolean("isregistered",false))

        finish();

    super.onRestart();

}

}

public class RegisterMbl extends AppCompatActivity {

    EditText txtph;

    DatabaseReference dref;

    String phone;

    ProgressDialog pbar;

    MsgDialogSetup msgDialogSetup;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_register_mbl);

        txtph=findViewById(R.id.txtphnum);


        msgDialogSetup=new MsgDialogSetup(this);

    }

    @SuppressWarnings("MissingSuperCall")

    @Override

    public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,
@NonNull int[] grantResults) {

```

```

        if(requestCode==100&&ActivityCompat.checkSelfPermission(this,
Manifest.permission.SEND_SMS)== PackageManager.PERMISSION_GRANTED)

            getOtp();

        else

            Toast.makeText(this, "Permission denied", Toast.LENGTH_SHORT).show();

    }

```

```

public void onClickGetOTP(View v){

    phone=txtph.getText().toString();

    Pattern pattern=Pattern.compile("[6,7,8,9]+[0-9]{9}");
    Matcher matcher=pattern.matcher(phone);
    if(!matcher.find()){

        msgDialogSetup.setupDialog("Enter valid phone number",Optional.empty());

    }

    else{

        pbar=ProgressDialog.show(this,"","Connecting ...");

        checkForNumberPresent();

    }

}

```

```

public void checkForNumberPresent(){

    dref= FirebaseDatabase.getInstance().getReference("ac_details");

    dref.addValueEventListener(new ValueEventListener() {

        boolean numfound=false;

        @Override

        public void onDataChange(@NonNull DataSnapshot snapshot) {

            for(DataSnapshot ac_no:snapshot.getChildren()){

```

```

        if(ac_no.child("phno").getValue().toString().equals(phone))
        {
            setLogin(phone);
            numfound=true;
            break;
        }
    }
    if(!numfound){
        pbar.hide();
        msgDialogSetup.setupDialog("Phone number is not registered",
Optional.empty());
    }
}

@Override
public void onCancelled(@NonNull DatabaseError error) {
}

});
}

public void setLogin(String ph){
    pbar.hide();

    dref = FirebaseDatabase.getInstance().getReference("RegisterInfo");
    dref.addListenerForSingleValueEvent(new ValueEventListener() {
        boolean exists=false;

        @Override
        public void onDataChange(@NonNull DataSnapshot snapshot) {
            for(DataSnapshot dshot:snapshot.getChildren()){
                if(dshot.getKey().equals(ph)){

```



```

        exists=true;

        break;
    }
}

if(!exists){

    if(ActivityCompat.checkSelfPermission(RegisterMbl.this,
android.Manifest.permission.SEND_SMS)== PackageManager.PERMISSION_GRANTED)

        getOtp();

    else{

        ActivityCompat.requestPermissions(RegisterMbl.this,new
String[] {Manifest.permission.SEND_SMS},100);

    }

}

else {

    pbar.hide();

    msgDialogSetup.setupDialog("Please    log    out    from    other    device",
Optional.empty());

}

}

@Override

public void onCancelled(@NonNull DatabaseError error) {

}

});

}

public void getOtp(){

    String otp= GenerateOTP.getOtp();

    pbar.hide();

```

```

SmsManager smsManager = SmsManager.getDefault();

ArrayList<String> parts = smsManager.divideMessage(otp + " " + " is ur otp");

smsManager.sendMultipartTextMessage(txtph.getText().toString(), "", parts, null, null);

Intent i=new Intent(RegisterMbl.this,OTP.class);

i.putExtra("phonenum",txtph.getText().toString());

i.putExtra("otp",otp);

startActivity(i);

finish();

}

@Override

public void onBackPressed() {

    msgDialogSetup.setupDialog("Registration is in Progress\nDo you want to quit",Optional.empty());

    msgDialogSetup.layoutDisc.setVisibility(View.VISIBLE);

    msgDialogSetup.txtsave.setText("Yes");

    msgDialogSetup.txtdiscard.setText("No");

    msgDialogSetup.layoutSave.setOnClickListener(v->{

        super.onBackPressed();

    });

    msgDialogSetup.layoutDisc.setOnClickListener(v->{

        msgDialogSetup.msg_dialog.hide();

    });

}

}

public class RegistrationCard extends AppCompatActivity {

    public EditText txtsecans,txtcardnum,txtcardvalid,txtcardpin ;

    public String[] qst={"What is the result of 2+2?","Which is the 2 digit\nmaximum number?","What is the result of 2x4x0?"};

```

```

public String[] ans={"4","99","0"};

public int qst_index=1,prevLength=0,length=0,prevLengthValid=0,lengthValid=0;

DatabaseReference dbase;

ActivityRegistrationCardBinding binding;

MsgDialogSetup msgDialogSetup;

@Override

protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    binding=ActivityRegistrationCardBinding.inflate(getLayoutInflater());

    setContentView(binding.getRoot());

    txtsecans=findViewById(R.id.txtsecans);

    txtcardnum=findViewById(R.id.txtcardnum);

    txtcardvalid=findViewById(R.id.txtcardvalid);

    txtcardpin=findViewById(R.id.txtcardpin);

    ArrayAdapter<String> adptr;

    adptr = new ArrayAdapter<String>(this, R.layout.spinner_layout2, qst);

    binding.spinnerQst.setAdapter(adptr);

    msgDialogSetup=new MsgDialogSetup(this);

    binding.txtcardnum.addTextChangedListener(new TextWatcher() {

        @Override

        public void beforeTextChanged(CharSequence s, int start, int count, int after) {

        }

        @Override

        public void onTextChanged(CharSequence s, int start, int before, int count) {

            if(binding.txtcardnum.getImeOptions()==EditorInfo.IME_ACTION_DONE)

                Toast.makeText(RegistrationCard.this, "done", Toast.LENGTH_SHORT).show();

```

```
        if(binding.txtcardnum.length()>prevLength && binding.txtcardnum.length()%5==0
&& binding.txtcardnum.length()!=0)
```

```
{
```

```
    String text,curText,lastText;
```

```
    length = binding.txtcardnum.length() - 1;
```

```
    prevLength=binding.txtcardnum.length();
```

```
    text = binding.txtcardnum.getText().toString();
```

```
    curText = text.substring(0, length);
```

```
    lastText = text.substring(length);
```

```
    binding.txtcardnum.setText(curText + " " + lastText);
```

```
    binding.txtcardnum.setSelection(binding.txtcardnum.length());
```

```
}
```

```
if(binding.txtcardnum.length()==19) {
```

```
    binding.txtcardvalid.requestFocus(View.FOCUS_FORWARD);
```

```
}
```

```
}
```

```
@Override
```

```
public void afterTextChanged(Editable s) {
```

```
    if(txtcardnum.length()==0) {
```

```
        prevLength=0;
```

```
    }
```

```
}
```

```
});
```

```
txtcardvalid.addTextChangedListener(new TextWatcher() {
```

```
    @Override
```

```

public void beforeTextChanged(CharSequence s, int start, int count, int after) {
}

@Override
public void onTextChanged(CharSequence s, int start, int before, int count) {
    String date=txtcardvalid.getText().toString();
    if(txtcardvalid.length()==3 && txtcardvalid.length()>prevLengthValid )
    {
        prevLengthValid=txtcardvalid.length();
        lengthValid = txtcardvalid.length() - 1;

txtcardvalid.setText(date.substring(0,lengthValid)+"/"+date.substring(lengthValid));
    }

    txtcardvalid.setSelection(txtcardvalid.length());

    if(binding.txtcardvalid.length()==5) {
        binding.txtcardpin.requestFocus(View.FOCUS_FORWARD);
    }
}

@Override
public void afterTextChanged(Editable s) {
    if(txtcardvalid.length()==0) {
        prevLengthValid=0;
    }
}

});

txtcardnum.setOnFocusChangeListener((v,hasfocus)->{
    if(!hasfocus&&txtcardnum.getText().length()<1) {

```

```

        msgDialogSetup.setupDialog("Please enter card number",Optional.empty());
    }

    else if(!hasfocus&&txtcardnum.getText().length()<19)

        msgDialogSetup.setupDialog("please enter valid card number",Optional.empty());
});

txtcardpin.addTextChangedListener(new TextWatcher() {

    @Override

    public void beforeTextChanged(CharSequence s, int start, int count, int after) {

    }

    @Override

    public void onTextChanged(CharSequence s, int start, int before, int count) {

    }

    @Override

    public void afterTextChanged(Editable s) {

        if(txtcardpin.length()==4)

            binding.txtsecans.requestFocus(View.FOCUS_FORWARD);

    }

});

}

public void onClickRegister(View v)

{

    String drnum=txtcardnum.getText().toString();

    String drvalid=txtcardvalid.getText().toString();

    String drpin=txtcardpin.getText().toString()

    if(drnum.length()==0)

        msgDialogSetup.setupDialog("Please enter card number",Optional.empty());

    else if(drnum.length()<19)

```

```

        msgDialogSetup.setupDialog("Please enter valid card number",Optional.empty());
else if(drvalid.length()==0)

        msgDialogSetup.setupDialog("Please enter card validity",Optional.empty());
else if(drpın.length()==0)

        msgDialogSetup.setupDialog("Please enter card pin",Optional.empty());
else if(binding.txtsecans.length()==0)

        msgDialogSetup.setupDialog("Please answer the security qst",Optional.empty());
else
if(!binding.txtsecans.getText().toString().equals(ans[binding.spinnerQst.getSelectedItemPosit
ion()])))

        msgDialogSetup.setupDialog("Wrong answer for qst", Optional.empty());
else {

        ProgressDialog pbar=ProgressDialog.show(this,"","\\nPlease Wait...");

        dbase = FirebaseDatabase.getInstance().getReference().child("ac_details");
        dbase.addValueEventListener(new ValueEventListener() {

            @Override

            public void onDataChange(@NonNull DataSnapshot snapshot) {

                Boolean found = false;

                for (DataSnapshot dshot : snapshot.getChildren()) {

                    if (dshot.child("drno").getValue().toString().equals(drnum)) {

                        if (dshot.child("drvalid").getValue().toString().equals(drvalid) &&

                            dshot.child("drpin").getValue().toString().equals(drpın)) {

                            checkLogin(dshot.child("phno").getValue().toString(),
dshot.child("name").getValue().toString(), dshot.getKey().toString());

                        } else

                            msgDialogSetup.setupDialog("Wrong Credential", Optional.empty());

                        found = true;

                    }

                }

```

```

    }

    if (!found)

        msgDialogSetup.setupDialog("Debit card not present", Optional.empty());

    pbar.hide();

    @Override

    public void onCancelled(@NonNull DatabaseError error) {

        }

    });

}

}

public void checkLogin(String ph,String name,String ac){

    DatabaseReference dref = FirebaseDatabase.getInstance().getReference("RegisterInfo");

    dref.addListenerForSingleValueEvent(new ValueEventListener() {

        boolean exists=false;

        @Override

        public void onDataChange(@NonNull DataSnapshot snapshot) {

            for(DataSnapshot dshot:snapshot.getChildren()){

                if(dshot.getKey().equals(ph)){

                    exists=true;

                    break;

                }

            }

            if(!exists){

                Intent i=new Intent(RegistrationCard.this, TpinMpin.class);

                i.putExtra("phone_no",ph);

                i.putExtra("ac_no",ac);

```



```

        i.putExtra("name",name);

        startActivity(i);

        finish();
    }

    else {

        msgDialogSetup.setupDialog("Please log out from other device",
Optional.empty());

    }

}

@Override

public void onCancelled(@NonNull DatabaseError error) {

}

});

}

@Override

public void onBackPressed() {

    msgDialogSetup.setupDialog("Registration is in Progress\nDo you want to
quit",Optional.empty());

    msgDialogSetup.layoutDisc.setVisibility(View.VISIBLE);

    msgDialogSetup.txtsave.setText("Yes");

    msgDialogSetup.txtdiscard.setText("No");

    msgDialogSetup.layoutSave.setOnClickListener(v->{

        super.onBackPressed();

    });

    msgDialogSetup.layoutDisc.setOnClickListener(v->{

        msgDialogSetup.msg_dialog.hide();

    });

}

```

```
}
```

Login Activity:

```
package com.example.gap;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.biometric.BiometricManager;
import androidx.biometric.BiometricPrompt;
import androidx.core.content.ContextCompat;
import androidx.core.content.IntentCompat;
import androidx.fragment.app.FragmentActivity;
import com.google.android.material.bottomsheet.BottomSheetDialog;
import com.google.firebase.auth.ActionCodeSettings;
import com.google.firebase.auth.AuthResult;
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;
import java.util.Calendar;
import java.util.Date;
import java.util.Optional;
import java.util.Timer;
import java.util.TimerTask;
import java.util.concurrent.Executor;
```

```

public class Login extends AppCompatActivity {

    BottomSheetDialog bs;

    TextView txtname,txtlog;

    FirebaseUser user=null;

    int cur_time,cur_min,hr,min;

    Calendar calendar=Calendar.getInstance();

    BiometricPrompt bioprompt;

    BiometricPrompt.PromptInfo promptInfo;

    ImageView fingerPrint;

    private VideoView videoView;

    private String videoUrl,phno,email,acno,ph;

    DatabaseReference dref;

    Vibrator v;

    MsgDialogSetup msgDialogSetup;

    int attempt;

    SharedPreferences u_details;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.logincreate);

        calendar.setTime(new Date());

        u_details=getSharedPreferences("user_details",MODE_PRIVATE);

        acno=u_details.getString("ac_no","");

        attempt=u_details.getInt("attempt",0);

        hr=u_details.getInt("hr",0);

        min=u_details.getInt("min",0);

```

```

ph=u_details.getString("phno","");

txtname=findViewById(R.id.txtname);

fingerPrint=findViewById(R.id.gifImageView4);

txtlog=findViewById(R.id.txtlogin);

SharedPreferences uref=getSharedPreferences("user_details",MODE_PRIVATE);

txtname.setText(uref.getString("name",""));

phno=uref.getString("phno","");

v= (Vibrator) getSystemService(Context.VIBRATOR_SERVICE);

msgDialogSetup=new MsgDialogSetup(this);

dref= FirebaseDatabase.getInstance().getReference("PinInfo");

attemptChecker();

```

```

FirebaseDatabase.getInstance().getReference("ac_details").child(acno).addListenerForSingleValueEvent(new ValueEventListener() {

```

```

    @Override

```

```

    public void onDataChange(@NonNull DataSnapshot snapshot) {

```

```

        email=snapshot.child("email").getValue().toString();

```

```

    }

```

```

    @Override

```

```

    public void onCancelled(@NonNull DatabaseError error) {

```

```

    }

```

```

});

```

```

txtlog.addTextChangedListener(new TextWatcher() {

```

```

    @Override

```

```

    public void beforeTextChanged(CharSequence s, int start, int count, int after) {

```

```

    }

```

```

    @Override

```

```

    public void onTextChanged(CharSequence s, int start, int before, int count) {

```

```

    }

    @Override

    public void afterTextChanged(Editable s) {

        if(txtlog.getText().length()==4){

            ProgressDialog dialog = new ProgressDialog(new
ContextThemeWrapper(Login.this, R.style.CustomFontDialog));

            dialog.setMessage("Logging in..");

            dialog.show();

            dref.child(phno).get().addOnCompleteListener(new
OnCompleteListener<DataSnapshot>() {

                @Override

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

                    if(task.isSuccessful()){

                        if(task.getResult().exists()){

if(task.getResult().child("LPin").getValue().equals(txtlog.getText().toString())){

                            Intent i=new Intent(Login.this,Home_Screen.class);

                            u_details.edit().putInt("attempt",0).commit();

                            dialog.dismiss();

                            startActivity(i);

                            finish();

                        }

                        else

                        {

                            txtlog.startAnimation(AnimationUtils.loadAnimation(Login.this,
R.anim.shake_animation));

                            txtlog.setText("");

                            if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {

```

```

        v.vibrate(VibrationEffect.createOneShot(500,
VibrationEffect.DEFAULT_AMPLITUDE));

        } else {

            v.vibrate(500);

        }

        dialog.dismiss();

        attempt++;

        u_details.edit().putInt("attempt",attempt).commit();

        attemptChecker();

        updateAttemptTime();

    }

}

}

}

});

}

}

});

}

```

@Override

```

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

    if(requestCode==AllConstants.EMAIL_VERIFY){

        FirebaseAuth.getInstance().signInWithEmailAndPassword(email,"1234567890")

        .addOnCompleteListener(new OnCompleteListener<AuthResult>() {

            @Override

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

                if(FirebaseAuth.getInstance().getCurrentUser().isEmailVerified()) {

```

```

        Dialog dialog=new Dialog(Login.this);

        dialog setContentView(R.layout.dialog_new_lpin);

        dialog.getWindow().setBackgroundDrawable(new
ColorDrawable(Color.TRANSPARENT));

        dialog.setCancelable(false);

        dialog.findViewById(R.id.lpin_close).setOnClickListener(r->{

            dialog.dismiss();

        });

        Button btnreset=dialog.findViewById(R.id.btnreset);

        TextView txtlnlpin,txtclpin;

        txtlnlpin=dialog.findViewById(R.id.txtlnlpin);

        txtclpin=dialog.findViewById(R.id.txtclpin);

        txtclpin.addTextChangedListener(new TextWatcher() {

            @Override

            public void beforeTextChanged(CharSequence s, int start, int count, int

after) {

            }

            @Override

            public void onTextChanged(CharSequence s, int start, int before, int

count) {

            }

            @Override

            public void afterTextChanged(Editable s) {

                if(s.length()>=4){

                    if(s.toString().equals(txtlnlpin.getText().toString()))

                        btnreset.setVisibility(View.VISIBLE);

```

```

        else
        {
            if (Build.VERSION.SDK_INT >=
Build.VERSION_CODES.O) {
                v.vibrate(VibrationEffect.createOneShot(500,
VibrationEffect.DEFAULT_AMPLITUDE));

txtclpin.startAnimation(AnimationUtils.loadAnimation(Login.this,
R.anim.shake_animation));

                s.clear();
            } else {
                v.vibrate(500);
            }
        }
    }
});

btnreset.setOnClickListener(r->{

    AlertDialog.Builder alert=new
AlertDialog.Builder(Login.this,R.style.CustomFontDialog);

    FirebaseDatabase.getInstance().getReference("PinInfo").child(ph).child("LPin")

        .setValue(txtclpin.getText().toString())

        .addOnSuccessListener(new OnSuccessListener<Void>() {

            @Override

            public void onSuccess(Void unused) {

                dialog.dismiss();

                alert.setMessage("Login pin updated successfully");

                alert.setPositiveButton("Ok",(k,l)->{

                    k.dismiss();

```



```

        });

        alert.show();

    }

})

.addOnFailureListener(new OnFailureListener() {

    @Override

    public void onFailure(@NonNull Exception e) {

        alert.setMessage(e.getMessage());

        alert.setPositiveButton("Dismiss", (k, l) -> {

            k.dismiss();

        });

        alert.show();

    }

});

});

FirebaseAuth.getInstance().getCurrentUser().delete();

dialog.show();

}

else

    Toast.makeText(Login.this, "Not verified",

        Toast.LENGTH_SHORT).show();

}

})

.addOnFailureListener(new OnFailureListener() {

    @Override

    public void onFailure(@NonNull Exception e) {

        Toast.makeText(Login.this, e.toString(), Toast.LENGTH_LONG).show();

```

```

        }

    });

}

super.onActivityResult(requestCode, resultCode, data);

}

public void onClickCalculator(View v)
{
    startActivity(new Intent(this, Calculator.class));
}

public void onClickFingerPrint(View v) {
    BiometricManager biometricManager = BiometricManager.from(this);
    switch (biometricManager.canAuthenticate()) {

        case BiometricManager.BIOMETRIC_ERROR_HW_UNAVAILABLE:

            Toast.makeText(this, "hardware not found", Toast.LENGTH_SHORT).show();

            break;

        case BiometricManager.BIOMETRIC_ERROR_NONE_ENROLLED:

            Toast.makeText(this, "hardware not enrolled", Toast.LENGTH_SHORT).show();

            break;

        case BiometricManager.BIOMETRIC_ERROR_NO_HARDWARE:

            Toast.makeText(this, "no hardware", Toast.LENGTH_SHORT).show();

            break;

        case BiometricManager.BIOMETRIC_SUCCESS:

            Executor executor = ContextCompat.getMainExecutor(this);

            bioprompt = new BiometricPrompt(Login.this, executor, new
            BiometricPrompt.AuthenticationCallback() {

                @Override

```

```

        public void onAuthenticationError(int errorCode, @NonNull CharSequence
errString) {
            super.onAuthenticationError(errorCode, errString);
        }

        @Override

        public void onAuthenticationSucceeded(@NonNull
BiometricPrompt.AuthenticationResult result) {
            super.onAuthenticationSucceeded(result);
            startActivity(new Intent(Login.this, Home_Screen.class));
            finish();
        }

        @Override

        public void onAuthenticationFailed() {
            attempt++;
            u_details.edit().putInt("attempt",attempt).commit();
            attemptChecker();
            updateAttemptTime();
            super.onAuthenticationFailed();
        }
    });

    promptInfo = new BiometricPrompt.PromptInfo.Builder().setTitle("Login using
Biometric")

        .setDeviceCredentialAllowed(true)

        .build();

    bioprompt.authenticate(promptInfo);
}
}

public void onClickHelp(View v)

```

```

{
    bs=new BottomSheetDialog(this);

    View z=getLayoutInflater().inflate(R.layout.activity_help_support,null,false);

    bs.setContentView(z);

    bs.show();

    bs.getWindow().clearFlags(WindowManager.LayoutParams.FLAG_DIM_BEHIND);

    bs.getWindow().setBackgroundDrawable(new
    ColorDrawable(Color.TRANSPARENT));

    bs.getWindow().getAttributes().windowAnimations = R.style.DialogAnimation;

    bs.getWindow().setGravity(Gravity.BOTTOM);

    bs.setCanceledOnTouchOutside(false);

    bs.findViewById(R.id.btnchatbot).setOnClickListener(new View.OnClickListener() {

        @Override

        public void onClick(View v) {

            startActivity(new Intent(Login.this,ChatBot.class));

        }

    });

    bs.findViewById(R.id.btnvideoguid).setOnClickListener(new View.OnClickListener() {

        @Override

        public void onClick(View v) {

            startActivity(new Intent(Login.this,Video_Player.class));

        }

    });

}

public void onClickFindAtm(View v){

    Toast.makeText(this, "came", Toast.LENGTH_SHORT).show();

}

```

```

void attemptChecker(){
    if(attempt<3){
        txtlog.setVisibility(View.VISIBLE);
        fingerPrint.setVisibility(View.VISIBLE);
        if(attempt!=0)
            msgDialogSetup.setupDialog("Wrong Pin\nYou have only "+(3-attempt)+"
attempts", Optional.of(R.drawable.info_icn));
    }
    else
    {
        calendar.setTime(new Date());
        cur_time=calendar.get(Calendar.HOUR_OF_DAY);
        cur_min=calendar.get(Calendar.MINUTE);

        hr=u_details.getInt("hr",0);
        min=u_details.getInt("min",0);

        if((cur_time==hr&&cur_min-min==0)||cur_time>=hr)
        {
            txtlog.setVisibility(View.VISIBLE);
            fingerPrint.setVisibility(View.VISIBLE);
            u_details
                .edit()
                .putInt("hr",0)
                .putInt("min",0)
                .putInt("attempt",0)

```

```

        .commit();
    }
    else {
        fingerPrint.setVisibility(View.INVISIBLE);
        txtlog.setVisibility(View.INVISIBLE);

        msgDialogSetup.setupDialog("Out of attempts\nWait for "+Math.abs(cur_time-
hr)+" hours", Optional.empty());
    }
}
}
}

void updateAttemptTime()
{
    if(attempt>=3){
        calendar.setTime(new Date());
        int time=calendar.get(Calendar.HOUR_OF_DAY)+3;
        if(time>=24)
            time-=24;
        int min=calendar.get(Calendar.MINUTE);

        u_details
            .edit()
            .putInt("hr",time)
            .putInt("min",min)
            .commit();
    }
}
}

```

```

public void onClickforgetPin(View v){

    Dialog dialog=new Dialog(this);

    dialog setContentView(R.layout.dialog_gmail_password);

    dialog.getWindow().setBackgroundDrawable(new
ColorDrawable(Color.TRANSPARENT));


    TextView txtemail;

    ProgressBar pbar=dialog.findViewById(R.id.progress);

    Button btn=dialog.findViewById(R.id.btnopenmail);

    txtemail=dialog.findViewById(R.id.txtemail);


    txtemail.setText(email);


    FirebaseAuth.getInstance().createUserWithEmailAndPassword(txtemail.getText().toString(),
"1234567890")

        .addOnSuccessListener(new OnSuccessListener<AuthResult>() {

            @Override

            public void onSuccess(AuthResult authResult) {

authResult.getUser().sendEmailVerification().addOnSuccessListener(new
OnSuccessListener<Void>() {

                @Override

                public void onSuccess(Void unused) {

                    user = authResult.getUser();

                    pbar.setVisibility(View.GONE);

                    btn.setVisibility(View.VISIBLE);

                }
            }
        })
    }
}

```

```

        }).addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                Toast.makeText(Login.this, "Failed",
                    Toast.LENGTH_SHORT).show();
            }
        });
    }
}

)

.addOnFailureListener(new OnFailureListener() {
    @Override
    public void onFailure(@NonNull Exception e) {
        Toast.makeText(Login.this, e.toString(), Toast.LENGTH_SHORT).show();
    }
});

btn.setOnClickListener(r->{
    dialog.dismiss();

    String url="https://mail.google.com/mail/";

    Intent i=new Intent(Intent.ACTION_VIEW);

    i.setData(Uri.parse(url));

    startActivityForResult(i, AllConstants.EMAIL_VERIFY);
})

dialog.show();
}
}

```


Calculator:

```
package com.example.gap;

import androidx.appcompat.app.AppCompatActivity;

import androidx.fragment.app.Fragment;

import androidx.fragment.app.FragmentManager;

import androidx.fragment.app.FragmentTransaction;

import android.os.Bundle;

import android.view.View;

public class Calculator extends AppCompatActivity {

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_calculator);

        replaceFragment(new fragment_loan("Personal","Housing"));

    }

    public void onClick(View v)

    {

        if(v.getId()==R.id.radioloan)

            replaceFragment(new fragment_loan("Personal","Housing"));

        else

            replaceFragment(new fragment_loan("RD","FD"));

    }

    private void replaceFragment(Fragment fragment) {

        FragmentManager fmanager=getSupportFragmentManager();

        FragmentTransaction ftransaction=fmanager.beginTransaction();

        ftransaction.replace(R.id.calculation_detail,fragment);

        ftransaction.commit();

    }

}
```

```

    }
}

package com.example.gap;

import static com.example.gap.misc.Decoration.setComma;

import android.annotation.SuppressLint;

import android.app.Activity;

import android.app.Dialog;

public class fragment_loan extends Fragment {

    View view;

    SeekBar skbar;

    TextInputEditText txtamt;

    Spinner spin;

    EditText txtintrst, txttenure;

    TextView txtemi,txtiamt,txttotal,txtvmore;

    double amt,roi,rate,emi,yearInt;

    double ten;

    RadioButton radpersonal,radhome;

    long min=50000;

    String name1,name2;

    String msg1[]={ "Invested amount","Est. return","Total amount"},

        msg2[]={ "EMI Amount","Interest","Total Amoount\nPayable"};

    Drawer draw;

    public fragment_loan(String s1,String s2) {

        name1=s1;

        name2=s2;

    }

    @Override

```

```

public View onCreateView(LayoutInflater inflater, ViewGroup container,
                           Bundle savedInstanceState) {

    view = inflater.inflate(R.layout.fragment_loan, container, false);

    skbar = view.findViewById(R.id.amtseekbar);

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

    spin = view.findViewById(R.id.spintenure);

    txtintrst = view.findViewById(R.id.txtinterest);

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

    txtvmore=view.findViewById(R.id.viewmore);

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

    radhome=view.findViewById(R.id.radhousing);

    radpersonal.setText(name1);

    radhome.setText(name2);

    txtamt.setFilters(new InputFilter[]{new InputFilterMinMax("1", "5000000")});

    String tenure[] = new String[]{"Months", "Years"};

    ArrayAdapter<String> adptr;

    adptr = new ArrayAdapter<String>(view.getContext(), R.layout.spinner_layout, tenure);

    spin.setAdapter(adptr);


    radpersonal.setOnClickListener(new View.OnClickListener() {

        @Override

        public void onClick(View v) {

            if (radpersonal.getText().toString().equals("Personal")) {

                setupTextBox(50000,5000000);

            }

            else

            {

```

```

        setupTextBox(25,20000000000);
    }
}
});

radhome.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (radhome.getText().toString().equals("Housing")) {
            setupTextBox(100000,1000000000);
        }
        else
        {
            setupTextBox(5000,10000000000);
        }
    }
});

txtamt.setOnFocusChangeListener(new View.OnFocusChangeListener() {
    @Override
    public void onFocusChange(View v, boolean hasFocus) {
        if(!hasFocus){
            try {
                if(Long.parseLong(txtamt.getText().toString())<min)
                    txtamt.setText(Long.toString(min));
            }
            catch (NumberFormatException e)
            {

```

```

        txtamt.setText(Long.toString(min));
    }
}
});

txtamt.addTextChangedListener(new TextWatcher() {

    @Override
    public void beforeTextChanged(CharSequence s, int start, int count, int after) {

    }

    @Override
    public void onTextChanged(CharSequence s, int start, int before, int count) {

    }

    @Override
    public void afterTextChanged(Editable s) {

    }
});

view.findViewById(R.id.btnmin).setOnClickListener(new View.OnClickListener() {

    @Override
    public void onClick(View v) {float interest =
Float.parseFloat(txtintrst.getText().toString());

        float inc = interest - 0.3f;

        txtintrst.setText(String.format("%.2f",inc));

    }

});

view.findViewById(R.id.btncalculate).setOnClickListener(new View.OnClickListener()
{

```

@Override

```
public void onClick(View v) {  
    amt=Float.parseFloat(txtamt.getText().toString());  
    roi=Float.parseFloat(txtintrst.getText().toString());  
    ten=Integer.parseInt(txttenure.getText().toString());  
    txtemi=view.findViewById(R.id.txtemi);  
    txtiamt=view.findViewById(R.id.txtiamt);  
    txttotal=view.findViewById(R.id.txttotal);  
    if(radhome.getText().toString().equals("FD")) {  
        RadioGroup rgrp=view.findViewById(R.id.radgrpfrgmnt);  
        if(rgrp.getCheckedRadioButtonId()==R.id.radpersonal)  
        {  
            if (spin.getSelectedItem().toString() == "Years") {  
                ten = ten * 12;  
                amt/=ten;  
            }  
            double n=amt*ten;  
            double est=amt*ten*(ten+1)*roi/100/24;  
            txtemi.setText(String.format("%.2f",n));  
            txtiamt.setText(String.format("%.2f",est));  
            txttotal.setText(String.format("%.2f",(n+est)));  
            draw=new Drawer(n,est,(n+est),msg1);  
        }  
        else {  
            if (spin.getSelectedItem().toString() == "Months")  
            {  
                ten = ten / 12;  
                double p = amt + (amt * ten * roi / 100);
```

```

        txtemi.setText(String.format("%.2f",amt));
        txtiamt.setText(String.format("%.2f",p-amt));
        txttotal.setText(String.format("%.2f",(p)));
        draw=new Drawer(amt,p-amt,p,msg1);
    }
    TextView txt;
    txt=view.findViewById(R.id.txtmsg1);
    txt.setText(msg1[0]);
    txt=view.findViewById(R.id.txtmsg2);
    txt.setText(msg1[1]);
    txt=view.findViewById(R.id.txtmsg3);
    txt.setText(msg1[2]);
}
else {
    if (spin.getSelectedItem().toString() == "Years")
        ten = ten * 12;
    rate = roi / 12 / 100;
    yearInt = rate * ten * amt;
    emi = amt * rate * (Math.pow((1 + rate), ten)) / (Math.pow((1 + rate), ten) - 1);
    txtemi.setText(String.format("%.2f",emi));
    txtiamt.setText(String.format("%.2f",yearInt));
    txttotal.setText(String.format("%.2f",(amt+yearInt)));
    TextView txt;
    txt=view.findViewById(R.id.txtmsg1);
    txt.setText(msg2[0]);
    txt=view.findViewById(R.id.txtmsg2);
    txt.setText(msg2[1]);
}

```

```

        txt=view.findViewById(R.id.txtmsg3);

        txt.setText(msg2[2]);

        draw=new Drawer(emi,yearInt,(amt+yearInt),msg2);

        InputMethodManager imm = (InputMethodManager)
        getContext().getSystemService(Activity.INPUT_METHOD_SERVICE);

        imm.hideSoftInputFromWindow(view.getWindowToken(), 0);
        view.findViewById(R.id.layoutloandetail).setVisibility(View.VISIBLE);

    }

});

view.findViewById(R.id.btnmax).setOnClickListener(new View.OnClickListener() {

    @SuppressWarnings("SetTextI18n")

    @Override

    public void onClick(View v) {

        float interest = Float.parseFloat(txtintrst.getText().toString());

        float inc = interest + 0.3f;

        txtintrst.setText(String.format("%.2f",inc));

    }

});

txtintrst.addTextChangedListener(new TextWatcher() {

    @Override

    public void beforeTextChanged(CharSequence s, int start, int count, int after) {

    }

    @Override

    public void onTextChanged(CharSequence s, int start, int before, int count) {

    }

    @Override

    public void afterTextChanged(Editable s) {

```



```

        minmax(txtintrst);
    }
});

spin.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {

    @Override

    public void onItemSelected(AdapterView<?> parentView, View selectedItemView, int
position, long id) {

        RadioGroup rgrp=view.findViewById(R.id.radgrpfrgmnt);

        RadioButton rbtn=view.findViewById(rgrp.getCheckedRadioButtonId());

        if(radpersonal.getText().toString().equals("RD")&&rbtn.isChecked())

            if(position==0)

                setupTextBox(25,2000000000);

            else

                setupTextBox(300,2000000000);

    }

    @Override

    public void onNothingSelected(AdapterView<?> parentView) {

    }

});

skbar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {

    @Override

    public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {

        txtamt.setText(Integer.toString(progress));

    }

    @Override

    public void onStartTrackingTouch(SeekBar seekBar) {

    }

}

```

```

        @Override

        public void onStopTrackingTouch(SeekBar seekBar) {

        }

    });

    txtvmore.setOnClickListener(new View.OnClickListener() {

        @Override

        public void onClick(View v) {

            draw.showDialog(getContext());

        }

    });

    return view;
}

public void setupTextBox(int min,int max){

    skbar.setMax(max);

    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {

        skbar.setMin(min);

    }

    txtamt.setText(Integer.toString(min));

    txtamt.setFilters(new InputFilter[]{new InputFilterMinMax(1, max)});

    view.findViewById(R.id.layoutloandetail).setVisibility(View.INVISIBLE);

    this.min = min;

}

public void minmax(EditText txt)

{

    if(txt.getText().toString().equals(""))

        txt.setText("0.1");

    float val=Float.parseFloat(txt.getText().toString());

```

```

        Log.d("text", ""+val);
        if(val>50.0f)
            txt.setText("50.0");
        else if(val<0.1f)
            txt.setText("0.1");
    }
}

```

Fund Transfer:

```

package com.example.gap;

import static com.example.gap.misc.GenerateOTP.getOtp;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import android.app.Activity;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import com.squareup.picasso.Picasso;
import java.util.ArrayList;
import java.util.Objects;

public class Fund_Transfer extends AppCompatActivity {

    ActivityFundTransferBinding binding;

    String phno,acno,name,imgurl,acno_sender,drpin,sPhno;

    double bal_sender,bal_recv;

    SharedPreferences u_details;

    public static Activity activity;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

```

```

super.onCreate(savedInstanceState);

binding=ActivityFundTransferBinding.inflate(getLayoutInflater());

setContentView(binding.getRoot());

activity=this;

phno=getIntent().getExtras().getString("phno");

acno=getIntent().getExtras().getString("acno");

u_details=getSharedPreferences("user_details",MODE_PRIVATE);

acno_sender=u_details.getString("ac_no","");

sPhno=u_details.getString("phno","");

```

```

FirebaseDatabase.getInstance().getReference("ac_details").child(acno_sender).addListenerForSingleValueEvent(new ValueEventListener() {

```

```

    @Override

```

```

    public void onDataChange(@NonNull DataSnapshot snapshot) {

```

```

        bal_sender=Double.parseDouble(snapshot.child("savings").getValue().toString());

```

```

        drpin= Objects.requireNonNull(snapshot.child("drpin").getValue()).toString();

```

```

    }

```

```

    @Override

```

```

    public void onCancelled(@NonNull DatabaseError error) {

```

```

    }

```

```

});

```

```

FirebaseDatabase.getInstance().getReference("ac_details").child(acno).child("savings").addListenerForSingleValueEvent(new ValueEventListener() {

```

```

    @Override

```

```

    public void onDataChange(@NonNull DataSnapshot snapshot) {

```

```

        bal_recv=Double.parseDouble(snapshot.getValue().toString());

```

```

    }

```

```

    @Override

```

```
        public void onCancelled(@NonNull DatabaseError error) {

        }

    });
```

```
FirebaseDatabase.getInstance().getReference("ac_details").child(acno).addValueEventListener(new ValueEventListener() {
```

```
    @Override

    public void onDataChange(@NonNull DataSnapshot snapshot) {

        binding.txtName.setText(snapshot.child("name").getValue().toString());

    }

    @Override

    public void onCancelled(@NonNull DatabaseError error) {

    }

});
```

```
FirebaseDatabase.getInstance().getReference("RegisterInfo").child(phno).addValueEventListener(new ValueEventListener() {
```

```
    @Override

    public void onDataChange(@NonNull DataSnapshot snapshot) {

        if(snapshot.hasChild("ImageUrl")){

            imgurl=snapshot.child("ImageUrl").getValue().toString();

            binding.iviewUser.setScaleType(ImageView.ScaleType.CENTER_CROP);

            Picasso.get().load(imgurl).into(binding.iviewUser);

        }

    }

    @Override

    public void onCancelled(@NonNull DatabaseError error)

    }
```

```

});

binding.txtAcno.setText(acno);

binding.imgBack.setOnClickListener(v->{

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

    finish();

});

binding.btnProceed.setOnClickListener(v->{

    String amt=binding.txtamt.getText().toString();

    Intent i=new Intent(this,Transaction_Pin.class);

    if(Long.parseLong(amt)>100000){

        String sec_pin=getOtp().substring(0,2);

        SmsManager smsManager = SmsManager.getDefault();

        ArrayList<String> parts =
smsManager.divideMessage(String.format(getResources().getString(R.string.otp_sms),sec_pi
n));

        smsManager.sendMultipartTextMessage(sPhno,"",parts,null,null);

        sec_pin=sec_pin+drpin.substring(2);

        i.putExtra("sec_pin",sec_pin);

    }

    i.putExtra("amt",amt);

    i.putExtra("ac_recv",acno);

    i.putExtra("phno",phno);

    i.putExtra("bal_recv",bal_recv);

    i.putExtra("bal_sender",bal_sender);

    startActivity(i);

});

}

}

```

```

public class FundTransferBeneficiary extends AppCompatActivity implements SelectInterface
{
    ActivityFundTransferBeneficiaryBinding binding;

    ArrayList<BeneficiaryModel> arrayList=new ArrayList<>();

    public static BeneficiaryAdapter adapter;

    SharedPreferences u_details;

    String acno;

    boolean hasBeneficiary;

    ProgressDialog progressDialog;

    DatabaseReference dref;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        binding=ActivityFundTransferBeneficiaryBinding.inflate(getLayoutInflater());

        setContentView(binding.getRoot());

        getSupportActionBar().setTitle("Fund Transfer");

        binding.recyclerViewBeneficiary.setLayoutManager(new LinearLayoutManager(this));

        u_details=getSharedPreferences("user_details",MODE_PRIVATE);

        acno=u_details.getString("ac_no","");

        adapter=new BeneficiaryAdapter(arrayList,this,acno,this);

        binding.recyclerViewBeneficiary.setAdapter(adapter);

        progressDialog=new ProgressDialog(new ContextThemeWrapper(this,
R.style.CustomFontDialog));

        progressDialog.setMessage("Loading...");

        progressDialog.show();

        dref=FirebaseDatabase.getInstance().getReference("Beneficiary").child(acno);

        dref.keepSynced(true);

        updateBeneficiary();
    }
}

```

```

    }

    public void updateBeneficiary(){

        dref.addListenerForSingleValueEvent(new ValueEventListener() {

            @Override

            public void onDataChange(@NonNull DataSnapshot snapshot) {

                for(DataSnapshot snapshot1:snapshot.getChildren()){

                    if(snapshot1.exists()&&snapshot1.hasChild("name")&&snapshot1.hasChild("ph")){

                        String ac_rec,name,ph,img_url=null;

                        ac_rec= snapshot1.getKey();

                        name=
Objects.requireNonNull(snapshot1.child("name").getValue()).toString();

                        ph= Objects.requireNonNull(snapshot1.child("ph").getValue()).toString();

                        if(snapshot1.hasChild("ImageUrl"))

                            img_url=Objects.requireNonNull(snapshot1.child("ImageUrl").getValue()).toString();

                        arrayList.add(new BeneficiaryModel(name,ph,ac_rec,img_url));

                    }

                }

                adapter.notifyDataSetChanged();

                progressDialog.dismiss();

                if(arrayList.size()>0)

                    binding.txtnobenifmsg.setVisibility(View.INVISIBLE);

                else

                    binding.txtnobenifmsg.setVisibility(View.VISIBLE);

            }

            @Override

            public void onCancelled(@NonNull DatabaseError error) {

```



```

        }

    });
}

@Override
protected void onRestart() {
    super.onRestart();
    arrayList.clear();
    updateBeneficiary();
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.add_benificiary_menu,menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    startActivity(new Intent(this,Add_Beneficiary_ac.class));
    return false;
}

@Override
public void onItemClick(int action_code,BeneficiaryModel beneficiaryModel) {
    if(action_code== AllConstants.DELETE){
        MsgDialogSetup dialog = new MsgDialogSetup(this);
        dialog.setupDialog("Do you want to remove " + beneficiaryModel.name + " from
beneficiary list?", Optional.empty());
        dialog.layoutDisc.setVisibility(View.VISIBLE);
        dialog.txtdiscard.setText("No");
    }
}

```

```

dialog.layoutDisc.setOnClickListener(v -> {
    dialog.msg_dialog.dismiss();
});
dialog.txtsave.setText("YES");
dialog.layoutSave.setOnClickListener(v -> {

```

```

FirebaseDatabase.getInstance().getReference("Beneficiary").child(acno).child(beneficiaryModel.ac).removeValue();

```

```

    MsgDialogSetup d = new MsgDialogSetup(this);
    d.setupDialog("Removed Successfully", Optional.empty());
    d.layoutSave.setOnClickListener(x -> {
        arrayList.remove(beneficiaryModel);
        if (arrayList.size() > 0)
            binding.txtnobenifmsg.setVisibility(View.INVISIBLE);
        else
            binding.txtnobenifmsg.setVisibility(View.VISIBLE);
        adapter.notifyDataSetChanged();
        d.msg_dialog.dismiss();
    });
    dialog.msg_dialog.dismiss();
});
}

else if(action_code==AllConstants.SEND){
    Intent i=new Intent(this,Fund_Transfer.class);
    i.putExtra("phno",beneficiaryModel.ph);
    i.putExtra("acno",beneficiaryModel.ac);
    startActivity(i);
    finish();
}

```

```

    }

}

public class Payment_QR extends AppCompatActivity {

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_payment_qr);

        replaceFragment(new Scan_QR(Payment_QR.this));

    }

    public void onClickScanQr(View v)

    {

    }

    private void replaceFragment(Fragment fragment) {

        FragmentManager fmanager=getSupportFragmentManager();

        FragmentTransaction ftransaction=fmanager.beginTransaction();

        ftransaction.add(R.id.qr,fragment);

        ftransaction.commit();

    }

}

public class Payement_Success extends AppCompatActivity {

    ActivityPayementSuccessBinding binding;

    MediaPlayer music;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        binding=ActivityPayementSuccessBinding.inflate(getLayoutInflater());

```

```

        setContentView(binding.getRoot());

        Animation animation= AnimationUtils.loadAnimation(getApplicationContext(),
            R.anim.scale_reduce);

        binding.llround.startAnimation(animation);

        music= MediaPlayer.create(this, R.raw.notification4);

        music.start();

        new Handler().postDelayed(new Runnable() {

            @Override

            public void run() {

                binding.lottiePay.setVisibility(View.VISIBLE);

                binding.lottiePay.playAnimation();

            }

        },3000);

    }

}

```

```

public class Transaction_Pin extends AppCompatActivity {

    ActivityTransactionPinBinding binding;

    SharedPreferences u_details;

    MsgDialogSetup msgDialogSetup;

    double amt_send,avail_bal_sender,avail_bal_recv;

    double updateAmt;

    double updateAmtRec;

    String ac_sender,sphno,rphno,ac_recv,sec_pin;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);
    }
}

```

```

binding=ActivityTransactionPinBinding.inflate(getLayoutInflater());

setContentView(binding.getRoot());

msgDialogSetup=new MsgDialogSetup(this);

u_details=getSharedPreferences("user_details",MODE_PRIVATE);

ac_sender=u_details.getString("ac_no","");

sphno=u_details.getString("phno","");
avail_bal_sender=Double.parseDouble(getIntent().getExtras().get("bal_sender").toString());

ac_rcv=getIntent().getExtras().get("ac_rcv").toString();

avail_bal_rcv=Double.parseDouble(getIntent().getExtras().get("bal_rcv").toString());

rphno=getIntent().getExtras().get("phno").toString();

);

shuffelBtn();
amt_send=Double.parseDouble(getIntent().getExtras().get("amt").toString());
});
}

public void onClickClear(View v){

String number=binding.txtPin.getText().toString();

int length=number.length()-1;

if(length>=0)

binding.txtPin.setText(number.substring(0,length));

else

shuffelBtn();

}

public void onClickPin(View v){

String text=binding.txtPin.getText().toString();

Button btn=(Button)v;

binding.txtPin.setText(text+btn.getText().toString());

}

```

```

void shuffelBtn(){

    Button button[]={binding.btn0,binding.btn1,binding.btn2,
        binding.btn3,binding.btn4,binding.btn5,binding.btn6,
        binding.btn7,binding.btn8,binding.btn9};

    ArrayList<Integer> gen_num=new ArrayList<>();

    Random random=new Random();

    for(int i=0;i<=9;i++){

        int num=random.nextInt(10);

        if(gen_num.contains(num)){

            i--;

            continue;

        }

        gen_num.add(num);

        button[i].setText(Integer.toString(num));

    }

}

@SuppressWarnings("SuspiciousIndentation")

public void onClickPay(View v){

    String tpin;

    if(amt_send>100000)

        tpin=getIntent().getExtras().get("sec_pin").toString();

    else

        tpin=u_details.getString("tpin","");

    //Toast.makeText(this, ""+tpin, Toast.LENGTH_SHORT).show();

    String eTpin=binding.txtPin.getText().toString();

    if(!tpin.equals("")){ //remove @ if error occurs

        if(tpin.equals(eTpin)){

```

```

        if(amt_send>avail_bal_sender)

            msgDialogSetup.setupDialog("Insufficient balance",Optional.empty());

        else

            {

                sendMoney();

            }

        }

        else{

            msgDialogSetup.setupDialog("Wrong Transaction Pin", Optional.empty());

        }

    }

    else

        msgDialogSetup.setupDialog("null tpin",Optional.empty());

}

void sendMoney(){

    ProgressDialog pbar=ProgressDialog.show(this,"","Transaction is in progress");

    String      timeStamp      =      new      SimpleDateFormat("dd-MM-yyyy
HH:mm:ss").format(Calendar.getInstance().getTime());

    String send_msg,recv_msg;

    updateAmt=avail_bal_sender-amt_send;

    updateAmtRec=amt_send+avail_bal_recv;

    send_msg=String.format(getResources().getString(R.string.bank_sms),ac_sender,"
debited",amt_send,timeStamp,updateAmt);

    recv_msg=String.format(getResources().getString(R.string.bank_sms),ac_recv,"
creadited",amt_send,timeStamp,updateAmtRec);

    FirebaseDatabase.getInstance().getReference("ac_details").child(ac_sender).child("savings").
setValue(updateAmt);

```

```
FirebaseDatabase.getInstance().getReference("ac_details").child(ac_recv).child("savings").setValue(updateAmtRec);
```

```
    updateTransaction();
```

```
    new Handler().postDelayed()->{
```

```
        SmsManager smsManager = SmsManager.getDefault();
```

```
        ArrayList<String> parts = smsManager.divideMessage(send_msg);
```

```
        smsManager.sendMultipartTextMessage(sphno,"",parts,null,null);
```

```
        ArrayList<String> parts2 = smsManager.divideMessage(recv_msg);
```

```
        smsManager.sendMultipartTextMessage(rphno,"",parts2,null,null);
```

```
        pbar.hide();
```

```
        Fund_Transfer.activity.finish();
```

```
        startActivity(new Intent(Transaction_Pin.this,Payment_Success.class));
```

```
        finish();
```

```
    },1000);
```

```
}
```

```
void updateTransaction(){
```

```
    String      timeStamp      =      new      SimpleDateFormat("dd-MM-yyyy HH:mm:ss").format(Calendar.getInstance().getTime());
```

```
    DatabaseReference dbref=FirebaseDatabase.getInstance().getReference("Transaction");
```

```
    String key=dbref.push().getKey();
```

```
    HashMap<String,Object> map=new HashMap<>();
```

```
    map.put("Description","Payment through mobile banking App");
```

```
    map.put("Date",timeStamp);
```

```
    map.put("Amount",amt_send);
```

```
    dbref.child(key).setValue(map);
```

```
    map.clear();
```



```

        map.put("Bal",updateAmt);

        map.put("Tdetails","Debited");

        map.put("ac",ac_recv);

        dbref.child(key).child(ac_sender).setValue(map);

        map.clear();

        map.put("Bal",updateAmtRec);

        map.put("Tdetails","Credited");

        map.put("ac",ac_sender);

        dbref.child(key).child(ac_recv).setValue(map);

    }
}

```

Help and Support:

```

package com.example.gap;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ProgressBar;
import android.widget.Spinner;
import android.widget.Toast;

public class ChatBot extends AppCompatActivity {

    RecyclerView rview;

    Spinner sp;

    ArrayList<ChatBotModalClass> arrayList=new ArrayList<>();

    ChatBotAdapter adapter;

```

```

ProgressBar pbar;

SharedPreferences sharedPreferences;

String image;

@Override

protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    setContentView(R.layout.activity_chat_bot);

    rview=findViewById(R.id.chat_container);

    rview.setLayoutManager(new LinearLayoutManager(this));

    arrayList.add(new ChatBotModalClass("", "Hi this is GAP ChatBot. I can help u out in
performing all the operations listed below"

        +" just select the query from the list and hit the send button", ""));

    adapter=new ChatBotAdapter(arrayList, this);

    rview.setAdapter(adapter);

    sharedPreferences=getSharedPreferences("user_details", MODE_PRIVATE);

    image=sharedPreferences.getString("image", "");

    sp=findViewById(R.id.helpspiner);

    pbar=findViewById(R.id.idPBLoading);

}

public void onClickChatSend(View v)

{

    pbar.setVisibility(View.VISIBLE);

    DatabaseReference
firebaseDatabase=FirebaseDatabase.getInstance().getReference("helper");

    String help_text=sp.getSelectedItem().toString();

    firebaseDatabase.addValueEventListener(new ValueEventListener() {

        @Override

        public void onDataChange(@NonNull DataSnapshot snapshot) {

```

```

        for(DataSnapshot dataSnapshot:snapshot.getChildren()){
            if(dataSnapshot.getKey().equals(help_text))
            {
                String ans_text=dataSnapshot.child("ans").getValue().toString();
                ans_text=ans_text.replaceAll("<>","\n");
                arrayList.add(arrayList.size(),new
ChatBotModalClass(help_text,ans_text,image));
                adapter.notifyDataSetChanged();
                rvview.scrollToPosition(adapter.getItemCount()-1);
                pbar.setVisibility(View.GONE);
                break;
            }
        }
    }

    @Override
    public void onCancelled(@NonNull DatabaseError error) {
        Toast.makeText(ChatBot.this, "cancelled", Toast.LENGTH_SHORT).show();
        pbar.setVisibility(View.GONE);
    }
});
}
}

public class Video_Player extends AppCompatActivity {
    ActivityVideoPlayerBinding binder;
    private MediaPlayer mediaPlayer;

    @Override
    protected void onCreate(Bundle savedInstanceState) {

```

```

super.onCreate(savedInstanceState);

binder=ActivityVideoPlayerBinding.inflate(getLayoutInflater());

setContentView(binder.getRoot());

binder.spinguid.setOnItemClickListener(new AdapterView.OnItemClickListener()
{
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id)
    {
        if(position%2==0)
            playVideo(R.raw.vid);
        else if(position==1)
            playVideo(R.raw.another);
        else
            playVideo(R.raw.blind_lights);
    }
    @Override
    public void onNothingSelected(AdapterView<?> parent) {
        Toast.makeText(Video_Player.this, "noting selected",
        Toast.LENGTH_SHORT).show();
    }
});
}

@Override
protected void onDestroy() {
    super.onDestroy();
    if (mediaPlayer != null) {
        mediaPlayer.release();
        mediaPlayer = null;
    }
}

```

```

    }
}

@Override
public boolean onTouchEvent(MotionEvent event) {
    binder.spinguid.setVisibility(View.VISIBLE);
    new Handler().postDelayed(new Runnable() {
        @Override
        public void run() {
            binder.spinguid.setVisibility(View.GONE);
            binder.videoView.setHasTransientState(true);
            binder.videoView.setTop(5);
        }
    },10000);
    return super.onTouchEvent(event);
}

public void playVideo(int id)
{
    String videoUrl="android.resource://" +getPackageName()+ "/" +id;
    Uri uri= Uri.parse(videoUrl);
    binder.videoView.setVideoURI(uri);
    MediaController mc=new MediaController(this);
    binder.videoView.start();
    binder.videoView.setMediaController(mc);
    mc.setAnchorView(binder.videoView);
}
}

```

```

public class ChatBotAdapter extends
RecyclerView.Adapter<ChatBotAdapter.ViewHolderClass>{

    ArrayList<ChatBotModalClass> arrayList;

    Context mContext;

    public ChatBotAdapter(ArrayList<ChatBotModalClass> arrayList, Context mContext) {

        this.arrayList = arrayList;

        this.mContext = mContext;

    }

    @NonNull

    @Override

    public ViewHolderClass onCreateViewHolder(@NonNull ViewGroup parent, int
viewType) {

        View
view=LayoutInflater.from(mContext).inflate(R.layout.layout_send_recieve,parent,false);

        ViewHolderClass viewhold=new ViewHolderClass(view);

        return viewhold;

    }

    @Override

    public void onBindViewHolder(@NonNull ViewHolderClass holder, int position) {

        if(arrayList.get(position).send.length(>0)

            holder.layout.setVisibility(View.VISIBLE);

        else

            holder.layout.setVisibility(View.GONE);

        holder.txtsend.setText(arrayList.get(position).send);

        holder.txtrev.setText(arrayList.get(position).recv);

        holder.ic_bot.setPadding(holder.ic_bot.getPaddingLeft(),60,holder.ic_bot.getPaddingRight(),
holder.ic_bot.getPaddingBottom());

```

```

        if(arrayList.get(position).img.equals(""))
            holder.img.setImageResource(R.drawable.user);
        else
            Picasso.get().load(arrayList.get(position).img).into(holder.img);
    }

    @Override
    public int getItemCount() {
        return arrayList.size();
    }

    public class ViewHolderClass extends RecyclerView.ViewHolder{
        TextView txtrev,txtsend;
        RelativeLayout layout;
        CardView ic_user,ic_bot;
        ImageView img;
        public ViewHolderClass(@NonNull View itemView) {
            super(itemView);
            txtrev= itemView.findViewById(R.id.txtrecieve);
            txtsend=itemView.findViewById(R.id.txtsend);
            layout=itemView.findViewById(R.id.layout_send);
            ic_user=(CardView) itemView.findViewById(R.id.icnu);
            ic_bot=itemView.findViewById(R.id.icnb);
            img=itemView.findViewById(R.id.sendImg);
        }
    }
}

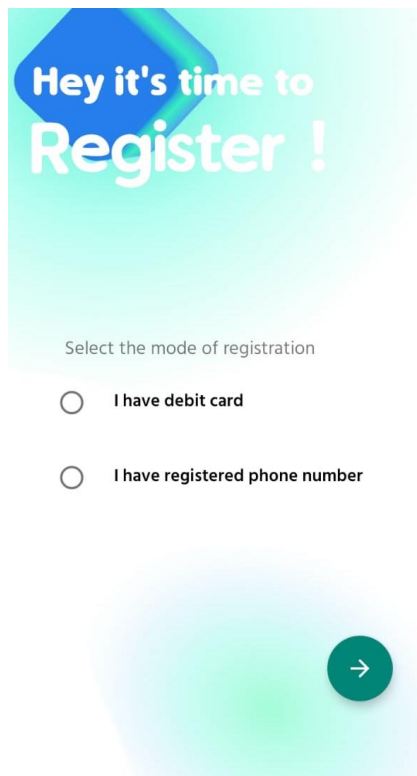
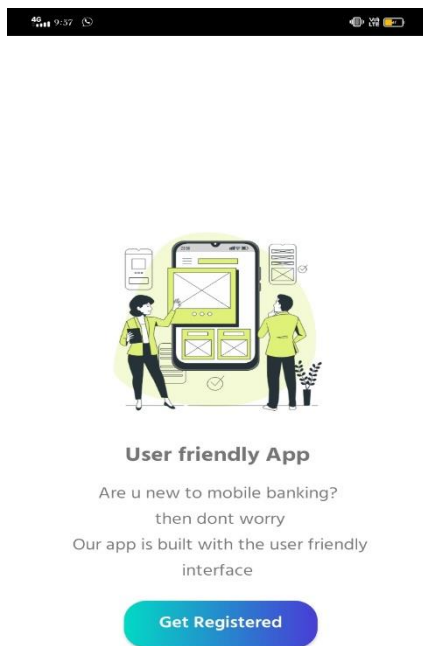
public class ChatBotModalClass {
    String send,recv,img;

```

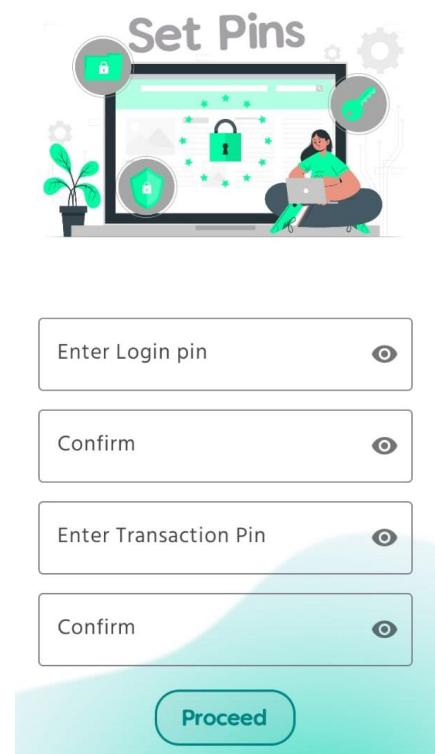
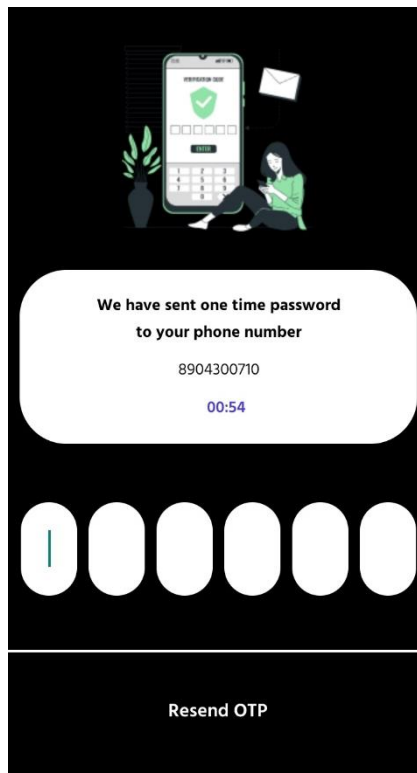
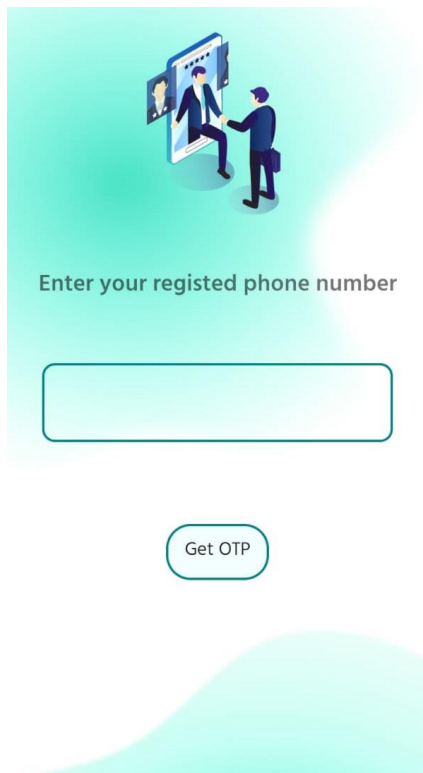
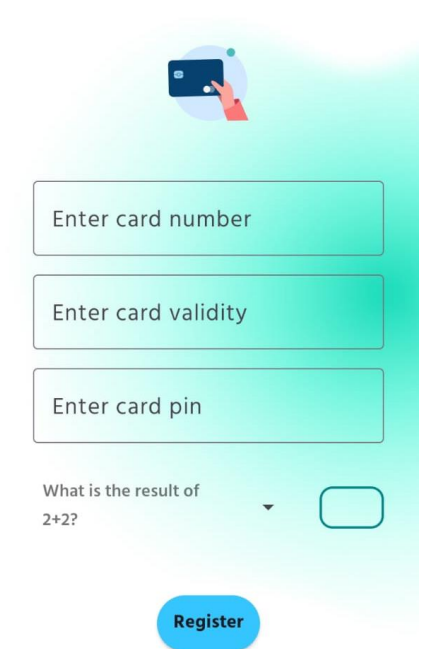
```
public ChatBotModalClass(String send, String recv,String img) {  
    this.send = send;  
    this.recv = recv;  
    this.img=img;  
}  
}
```


6. User Interfaces

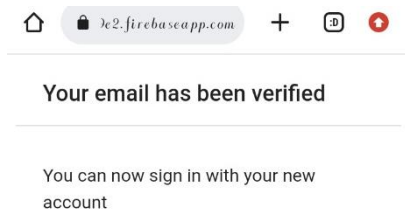
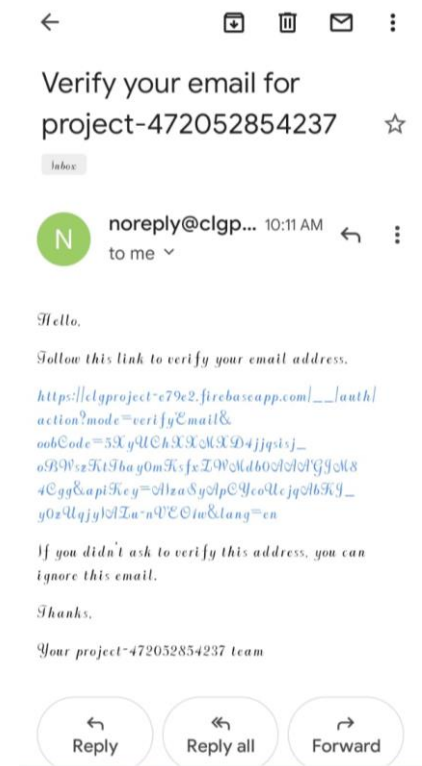
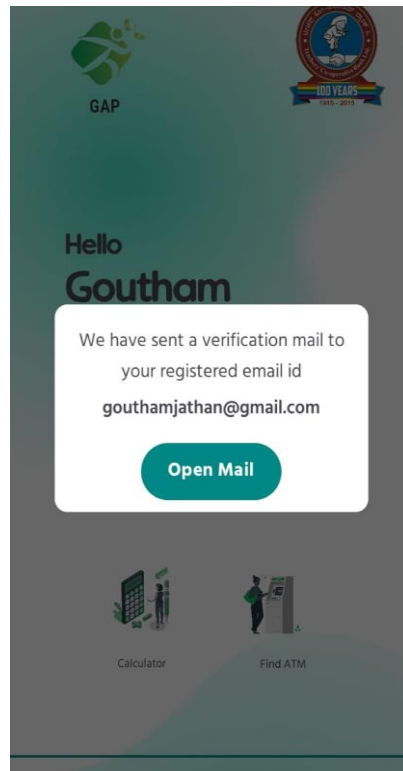
Registration screen



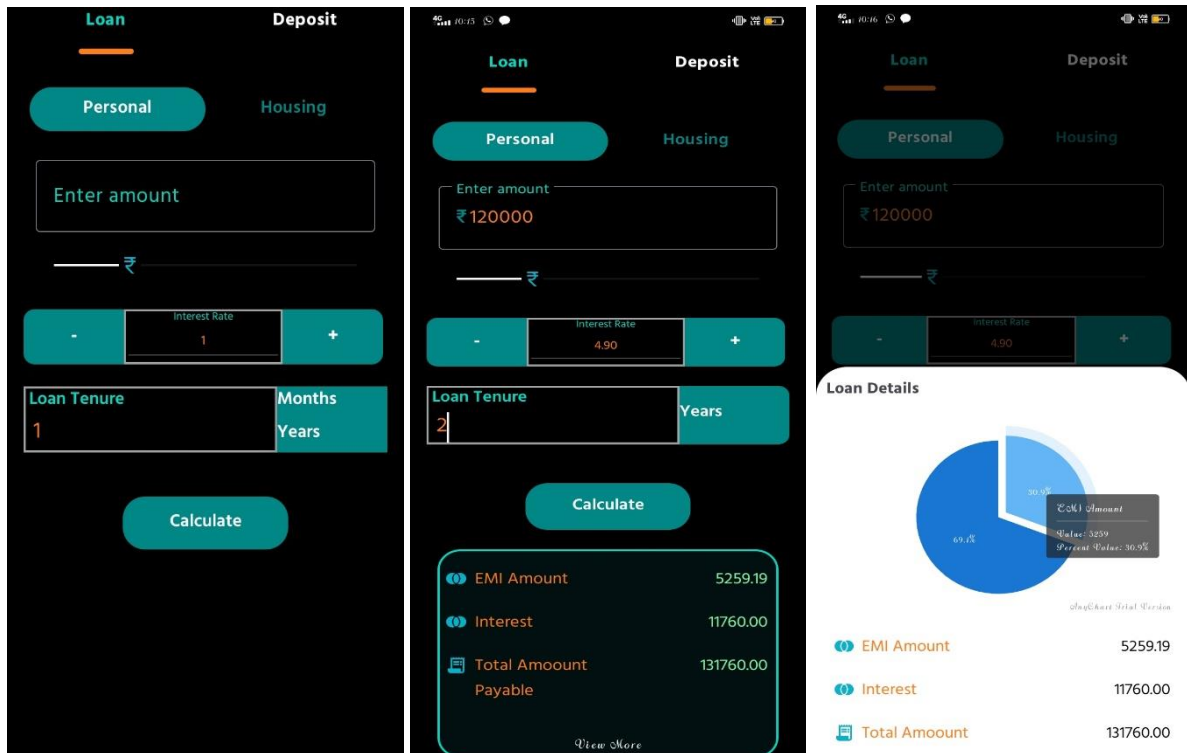
Enter your card details



Login Screen



Calculator



E-Passbook

Name	Goutham
Account No	12345
ISFC Code	TAB2023101
Branch	Santhekatte branch

Account No	ISFC Code
67434	TAB2023

Date	Debited	Balance
06-07-2023 12:27:06	- ₹ 1,00,001	₹ 8,44,948

Description Payment through mobile banking App		
---	--	--

Account No	ISFC Code
67434	TAB2023

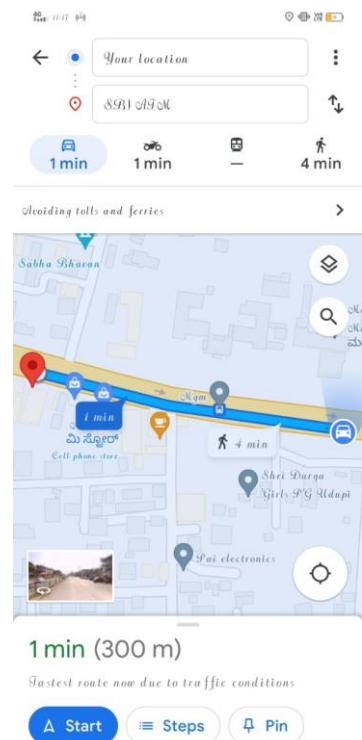
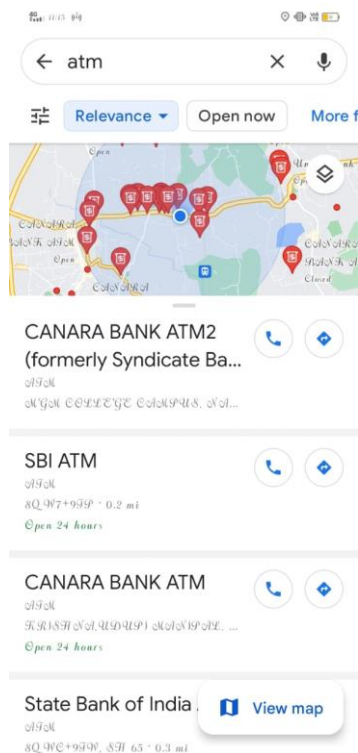
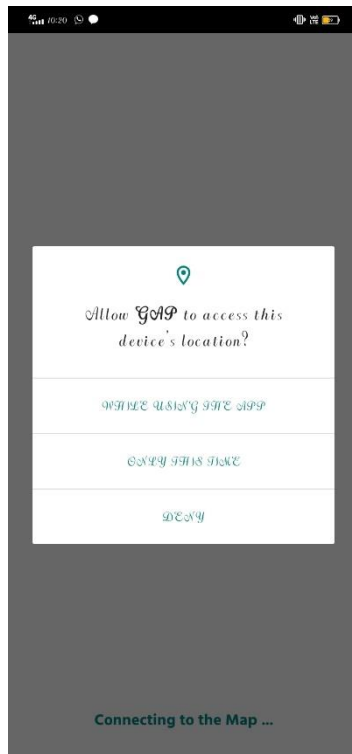
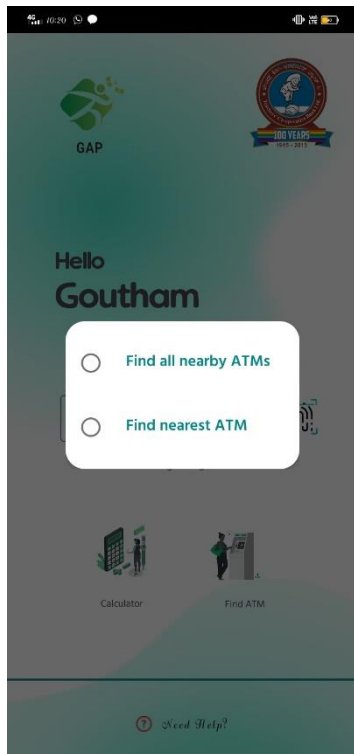
Date	Credited	Balance
06-07-2023 22:37:33	+ ₹ 20	₹ 8,44,968

Description Payment through mobile banking App		
---	--	--

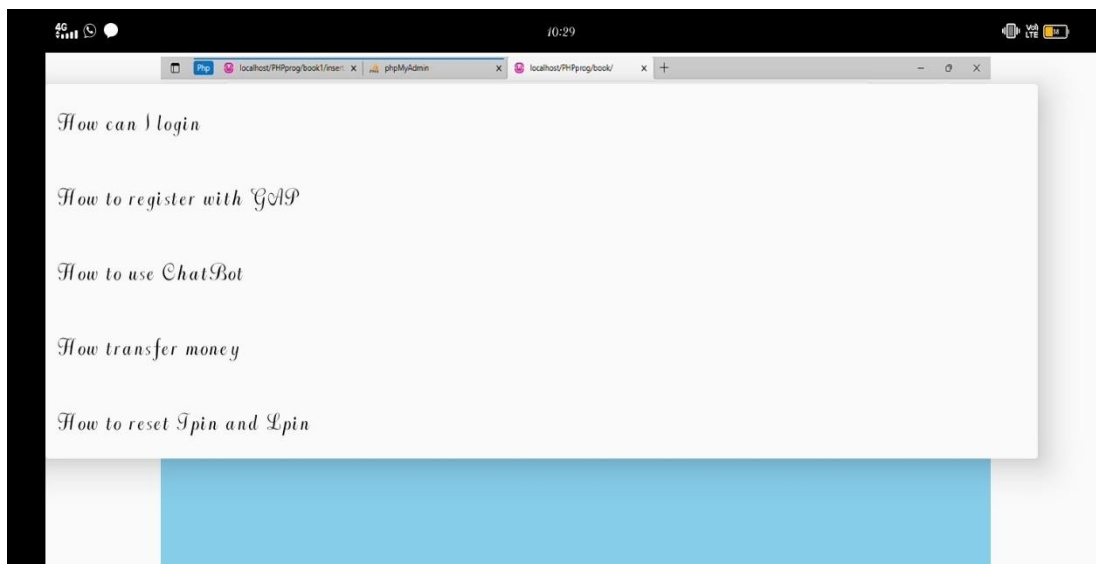
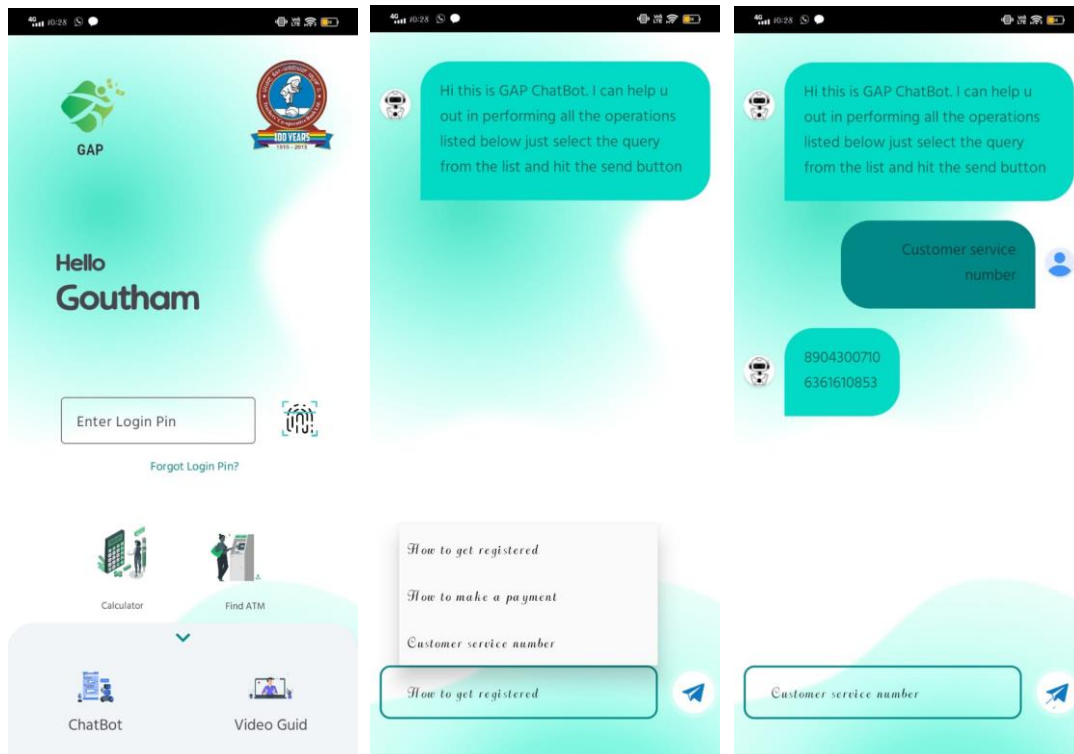
Account No	ISFC Code
54321	TAB2023

Date	Debited	Balance
08-07-2023 10:55:41	- ₹ 10	₹ 8,44,958

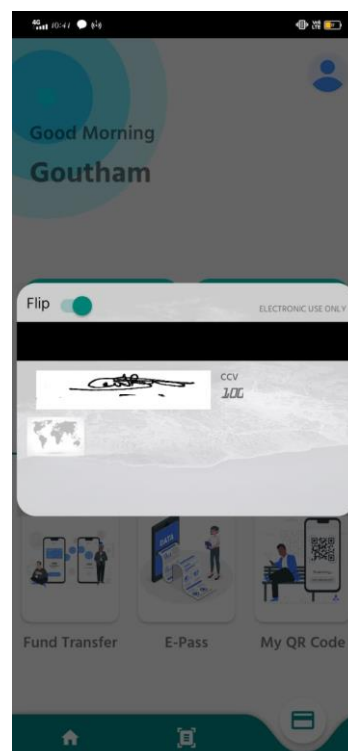
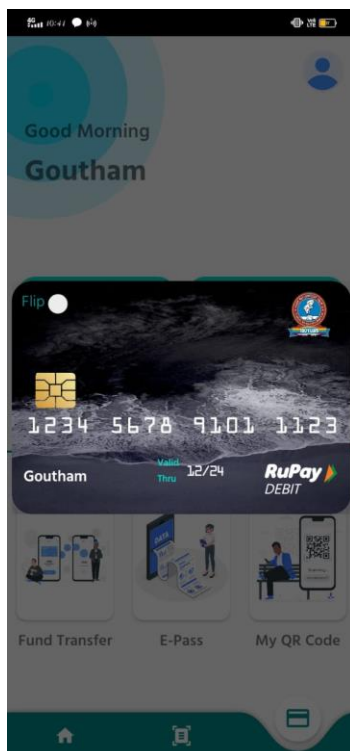
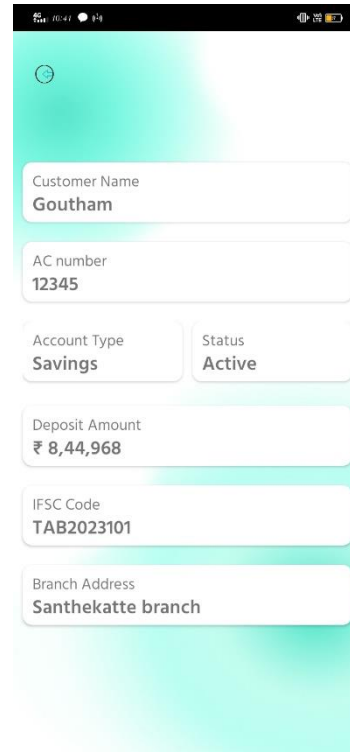
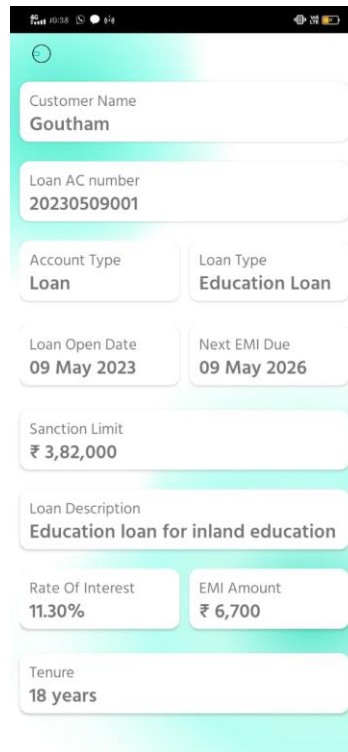
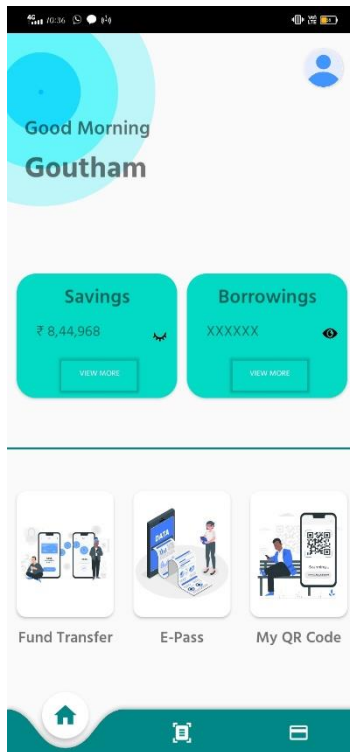
Search ATM



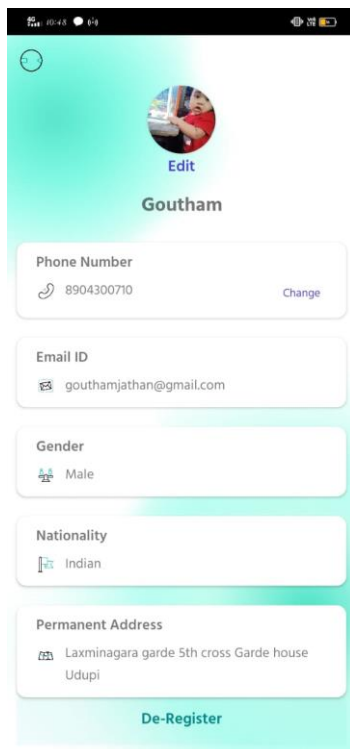
Help and Support



Home Screen



User Profile



Mobile app screenshot showing the user profile form in a light theme. The form includes fields for Phone Number, Email ID, Gender, Nationality, and Permanent Address, each with an 'Edit' button. A 'De-Register' button is at the bottom.

Phone Number
8904300710 [Change](#)

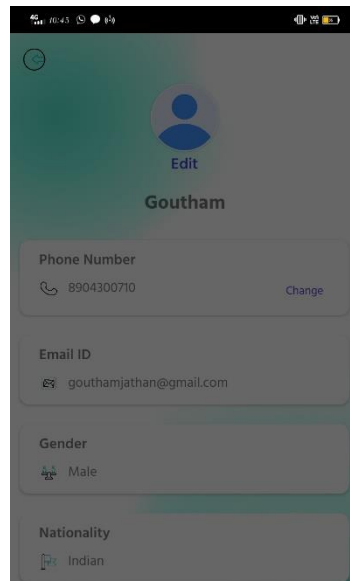
Email ID
gouthamjathan@gmail.com

Gender
Male

Nationality
Indian

Permanent Address
Laxminagara garde 5th cross Garde house
Udupi

[De-Register](#)



Mobile app screenshot showing the user profile form in a dark theme. The form includes fields for Phone Number, Email ID, Gender, Nationality, and Permanent Address, each with an 'Edit' button.

Phone Number
8904300710 [Change](#)

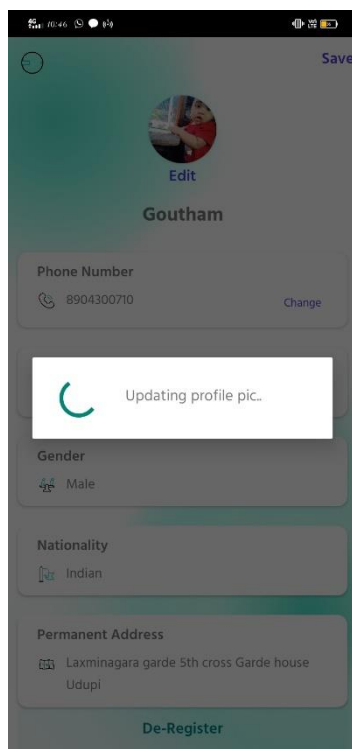
Email ID
gouthamjathan@gmail.com

Gender
Male

Nationality
Indian

 [Change Photo](#)

 [Remove Photo](#)



Mobile app screenshot showing the user profile form in a dark theme. A progress bar and 'Updating profile pic..' message are displayed over the form.

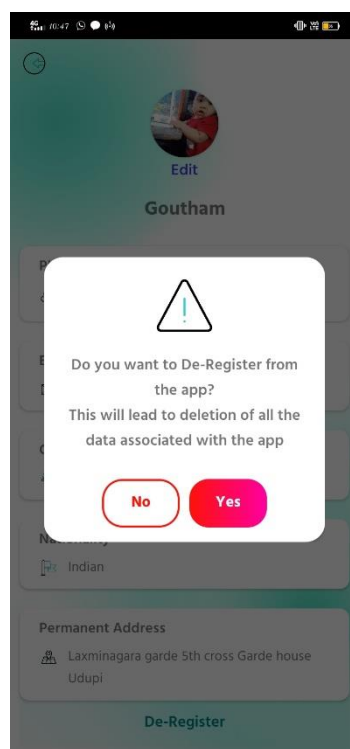
Phone Number
8904300710 [Change](#)

Gender
Male

Nationality
Indian

Permanent Address
Laxminagara garde 5th cross Garde house
Udupi

[De-Register](#)



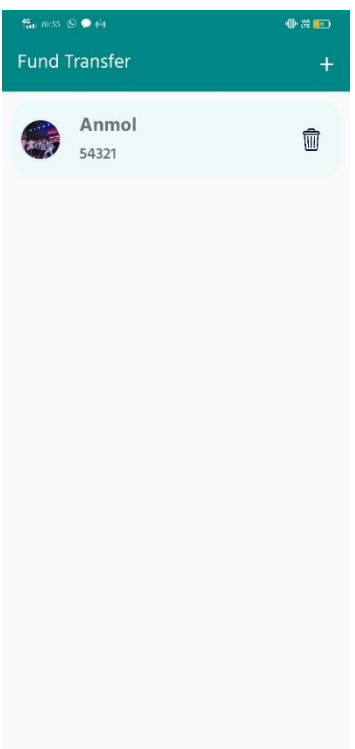
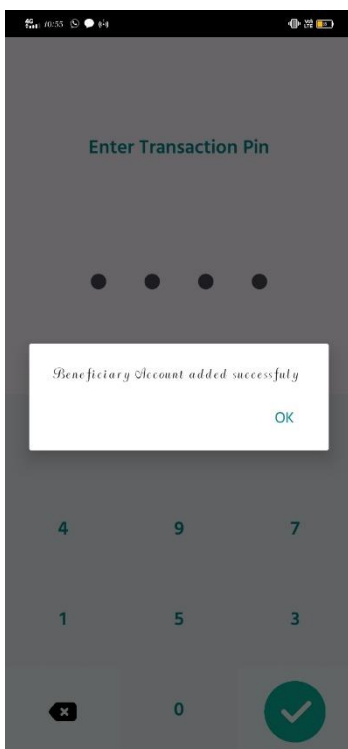
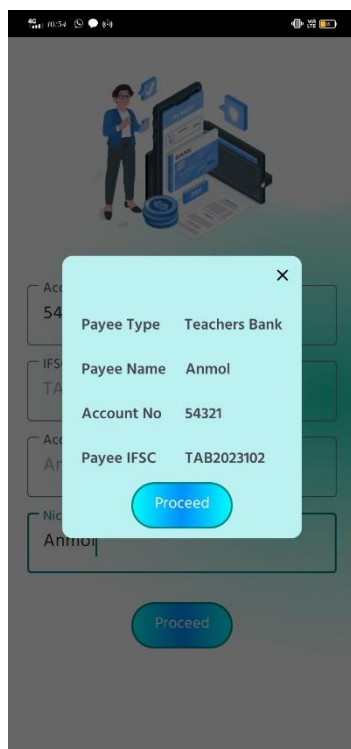
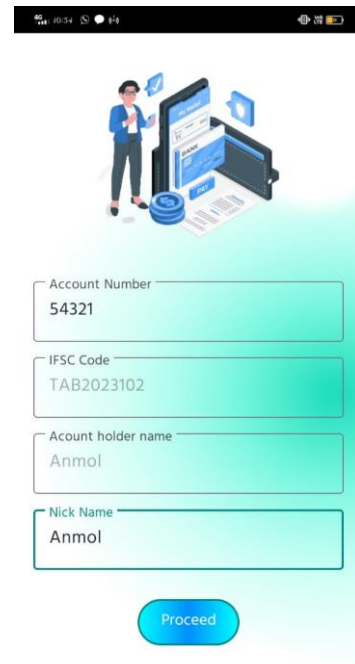
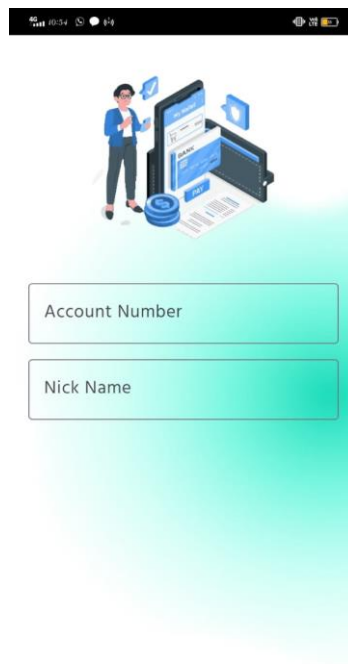
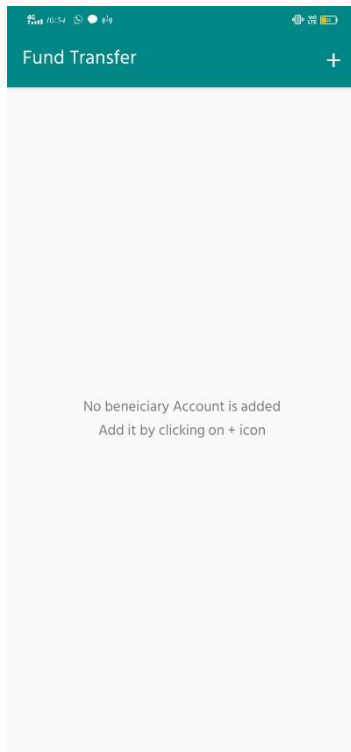
Mobile app screenshot showing the user profile form in a dark theme. A confirmation dialog is displayed over the form, asking if the user wants to de-register.

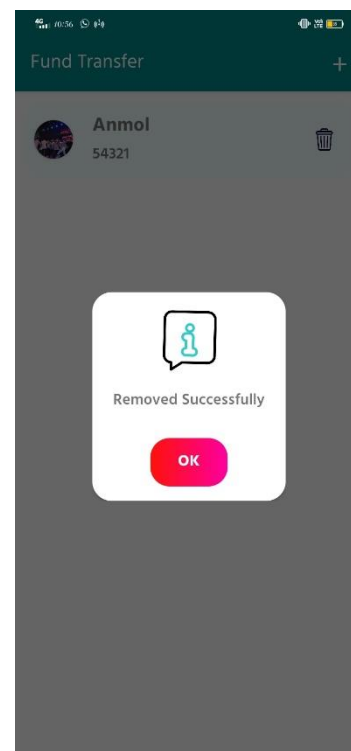
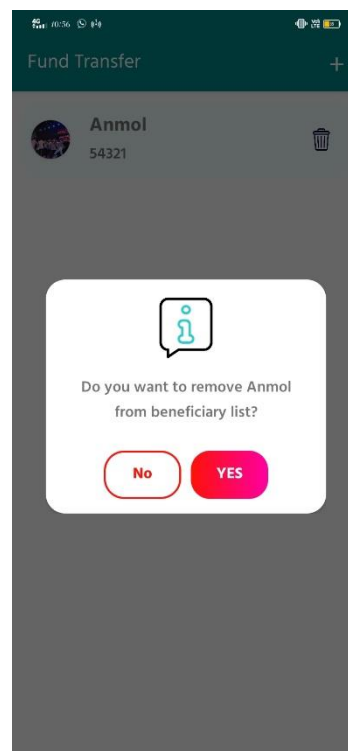
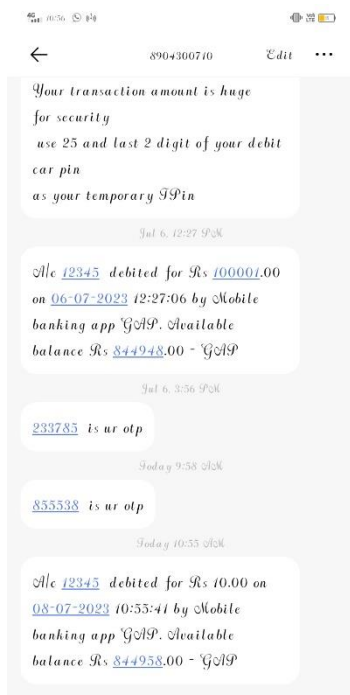
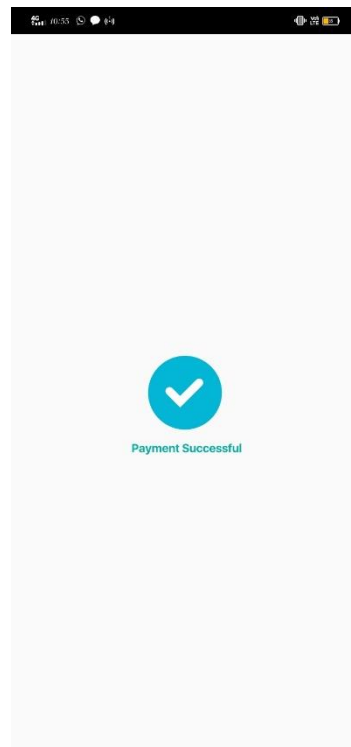
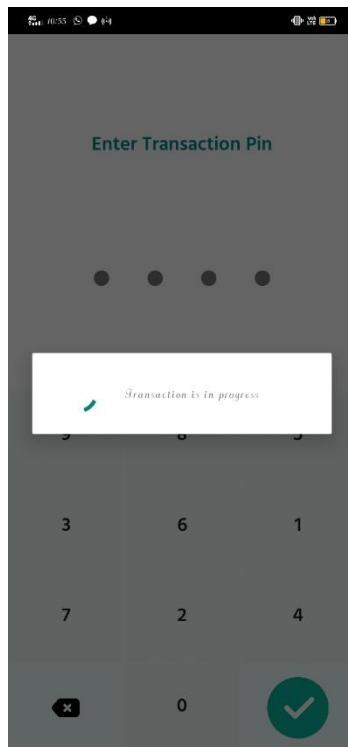
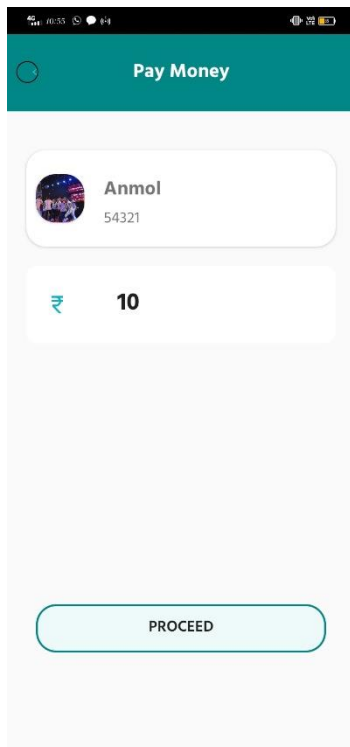
Do you want to De-Register from the app?
This will lead to deletion of all the data associated with the app

[No](#) [Yes](#)

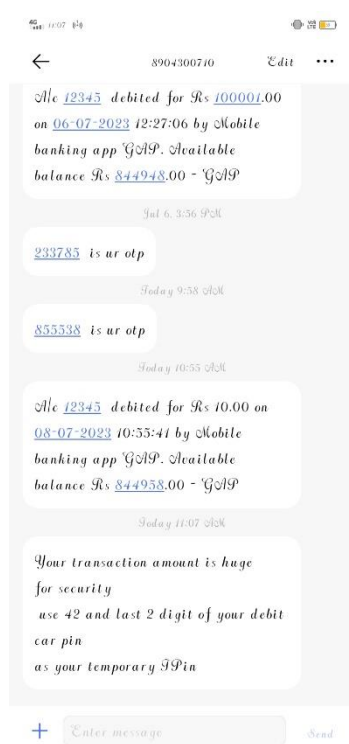
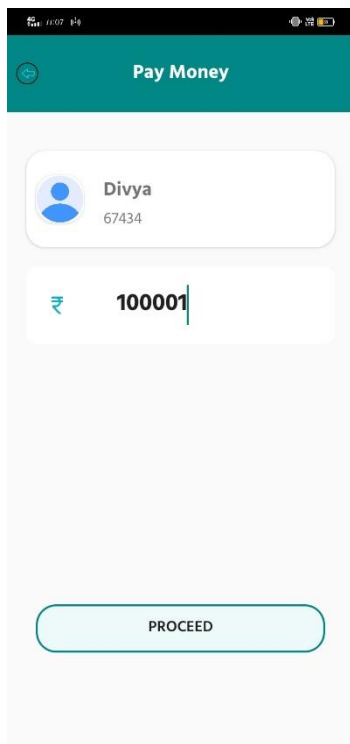
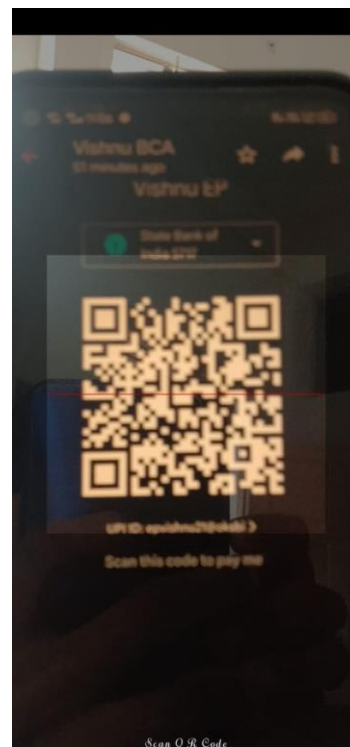
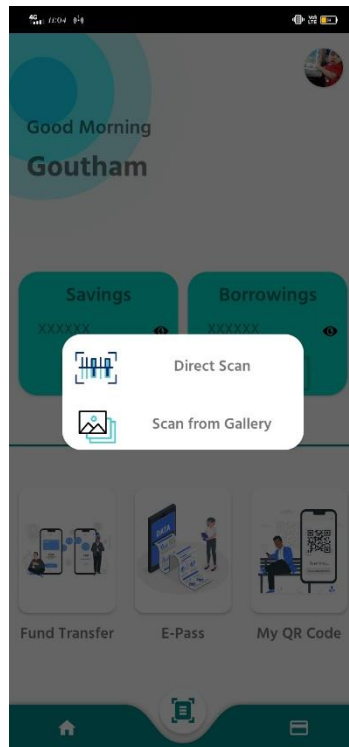
[De-Register](#)

Fund Transfer





QR Code Scan



7. Testing

i.Introduction:

Testing and implementation are the process, which tells the reality efficiency and the flexibility of the system design. Reliability means how much the user is expecting from the system. Flexibility tells how much the user is comfortable and hopes additional facilities with the system.

It is the vital to the success of the system. System testing makes the logical assumption that if all the part of the system is correct, the goal will be successively achieved. It is a critical element of software quality assurance and represents the ultimate review if specification design and coding.

Testing presents interesting anomaly of the software. The testing phase involves testing system using various test data. The first test of the system to see whether it produces the correct output. When the software is tested the actual output is tested with the expected output. If there is discrepancy the sequence of instruction must be traced to determine the problem. Breaking the program down into self-contained portions, each of which can be checked at certain key points facilitates the process.

The best program is worthless if it does not meet needs. The first step in system testing is to prepare a plan that will test all aspects of the system in a way that promotes its credibility among potential users. The development of software system involves a series of production activities where opportunities for injection of human error are enormous. Error may occur at the very imperfectly specified as well as later design and development stages.

ii. Test Reports:

a. Unit Testing:

It focuses on verification efforts on the smallest unit of software design module. Using the unit test plans, prepared in the design phase of the system as guide important control paths are tested to uncover error within the boundary of the module. The interfaces of each of the module under consideration are tested. Boundary condition was checked. Each unit was thoroughly tested to check if it might fail in any possible situation. This testing was carried out during programming itself.

b.Integration Testing:

Data can lose across an interface one module can have an adverse effect on another sub function, when combined may not produce the desired function. Global data structures can present problems. It is symmetric testing for constructing test to unrecovered error associated with the interface. All modules are combined in these testing process. Then the entire program was tested as a whole.

c.System Testing:

Software testing is a critical element of software quantity assurance and represents the ultimate review of specifications. design, and coding. The testing phase involves the testing of system using various test data. Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested. Those test data, error was found and corrected by using some testing steps. Thus, a series testing is performed on the system before it is ready for implementation.

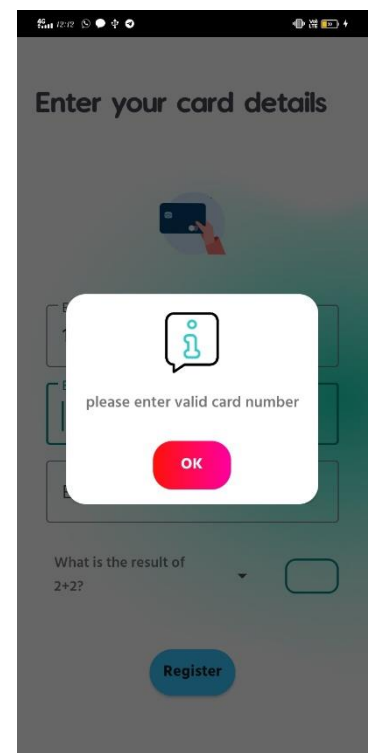
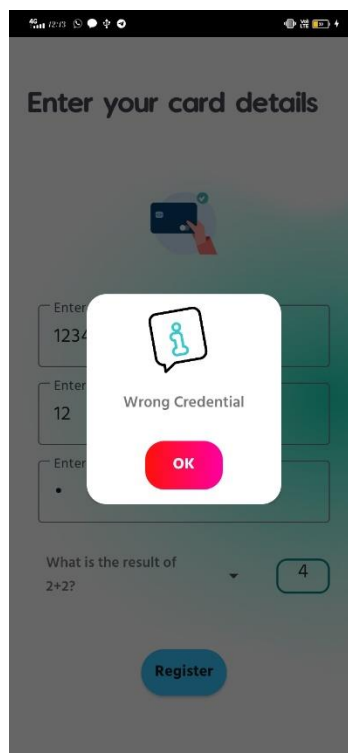
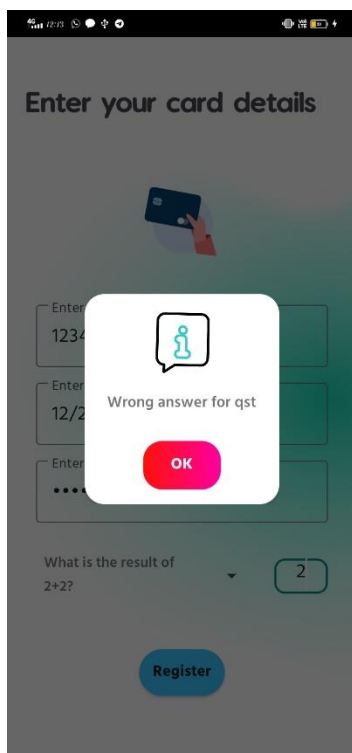
iii. Test Plan:

Test plan	Test case
1	Test for debit card details.
2	Test for user registered phone number
3	Test for setting transaction and login pin
4	Test for login authentication
5	Test for fund transfer

iv. Test Cases:

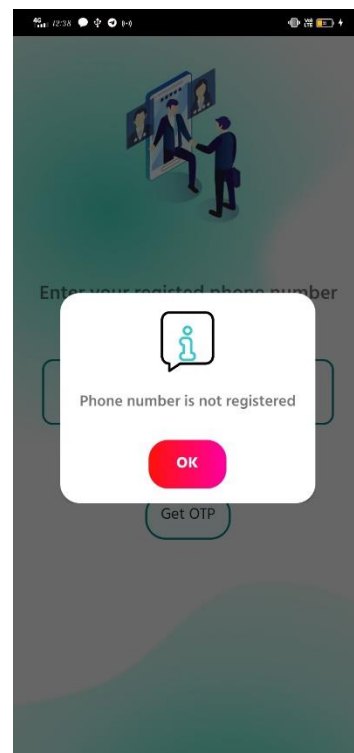
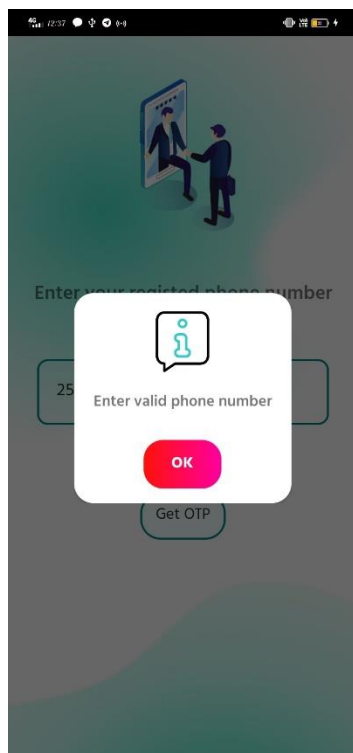
Test Case: 1

- Objectives:
Test for debit card details.
- Test Data
 - Valid: 16-digit valid debit card number, 4-digit valid date and 3 digit ccv with right answer to security question
 - Invalid:
 - Invalid card details
 - Wrong answer to security question
- Result:
 - Valid: User will be registered successfully and directed to set Transaction and Login pin.
 - Invalid: Proper error message will be displayed and allow the user to re-enter the card details.
- Conclusion:
Both the valid and invalid result is tested. Output matches with the required result. Hence Test case is successful.



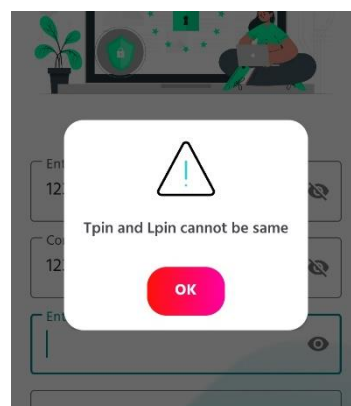
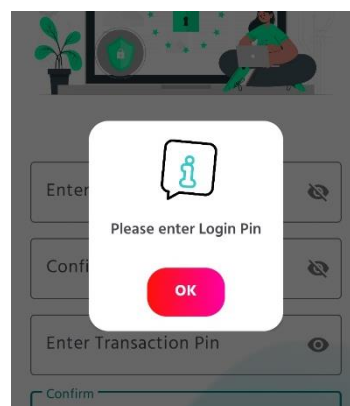
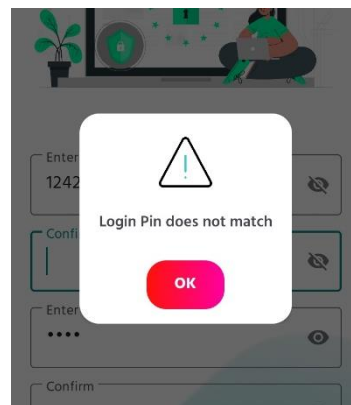
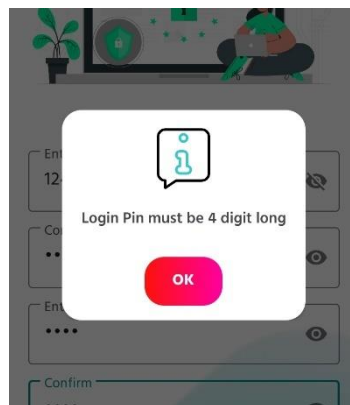
Test Case: 2

- Objectives:
Test for user registered phone number
- Test Data
 - Valid:
10-digit Indian phone number starting with 6,7,8 or 9. It should be linked to the bank account.
 - Invalid:
 - More or less than 10 digits
 - 10-digit number that is not starting with 6,7,8,9
 - Unlinked phone number
 - Empty phone number field
- Result:
 - Valid: A 6-digit OTP will be sent to given phone number and allow the user to enter OTP.
 - Invalid: Proper error message will be displayed and allow the user to re-enter phone number.
- Conclusion:
Both the valid and invalid result is tested. Output matches with the required result. Hence Test case is successful.



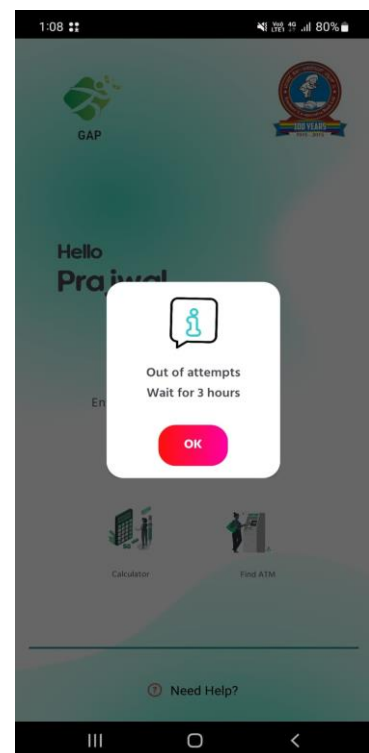
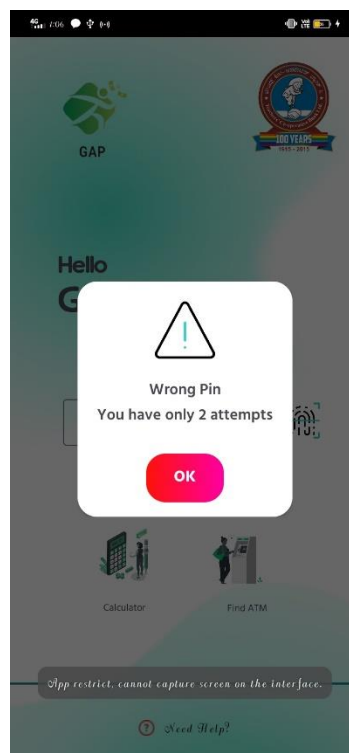
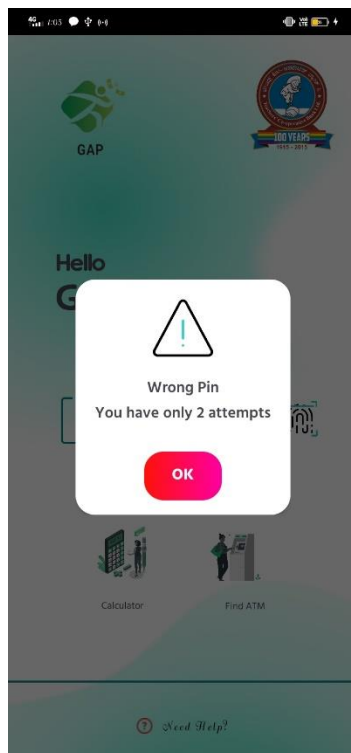
Test Case: 3

- Objectives:
Test for setting transaction and login pin
- Test Data:
 - Valid:
It should be a 4-digit pin. Transaction and login pin should be unique
 - Invalid:
 - Less than 4 digits
 - Transaction and login pin are same
 - Empty Transaction and login pin field
 - Mismatching pin and confirm pin
- Result:
 - Valid: User will be registered successfully and directed to login page
 - Invalid: Proper error message will be displayed and allow the user to re-enter transaction and login pin.
- Conclusion:
Both the valid and invalid result is tested. Output matches with the required result. Hence Test case is successful.



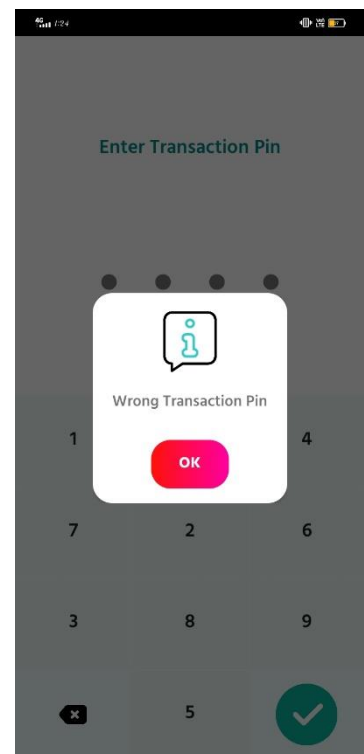
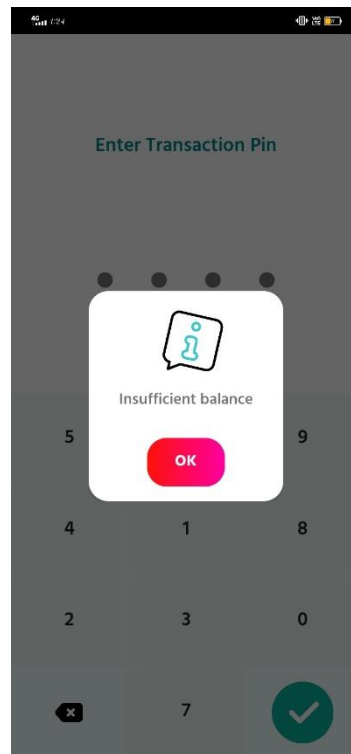
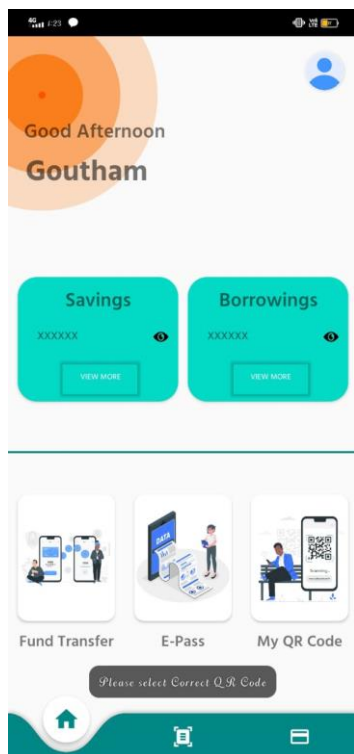
Test Case: 4

- Objectives:
Test for login authentication
- Test Data:
 - Valid:
 - 4-digit valid login pin
 - Authenticated users finger print or facial pattern
 - Invalid:
 - Wrong login pin
 - Mismatching user finger print or facial pattern
- Result:
 - Valid: User will logged in successfully and directed to home screen
 - Invalid: Proper error message will be displayed and chance will be deducted from attempts
- Conclusion:
Both the valid and invalid result is tested. Output matches with the required result. Hence Test case is successful.



Test Case: 5

- Objectives:
Test for fund transfer.
- Test Data
 - Valid:
 - Valid QR code
 - Correct transaction pin
 - Transfer amount should be lessor than available money
 - Invalid:
 - Invalid QR Code
 - Wrong transaction pin
 - Transfer money greater than available money
- Result:
 - Valid:
Payment will successful. Amount will be debited and credited from and to senders account and receivers account respectively
 - Invalid:
Proper error message will be displayed and transaction will be rolled back.
- Conclusion:
Both the valid and invalid result is tested. Output matches with the required result. Hence Test case is successful.



8.Limitation

- a. App does not support in adding multiple accounts under one login
- b. App does not support transfer of fund between 2 different banks
- c. To verify the email, Email should be opened within the app.
- d. It does not support cross platform.

9.Scope for Enhancement

- a. Allowing user to add multiple Bank account
- b. Adding additional modules that support fund transfer between 2 or more different bank
- c. Allowing the users to make bill and loan payments

10.Abbreviation

SRS : Software Requirement Specification.

OS : Operating System

SQL : Structured Query Language.

DFD : Data Flow Diagram

CFD : Context Flow Diagram

JSON: Java Script Object Notation

QR: Quick Response

IDE: Integrated Development Environment

11.Bibliography

The content for this project has been taken from the following sources:

www.stackoverflow.com

www.developer.android.com

THANK YOU