

Team ID: 591022

Project: Snack Squad

Performance and Final Submission Phase

Index

1. **INTRODUCTION**
 - 1.1 Project Overview
 - 1.2 Purpose
2. **LITERATURE SURVEY**
 - 2.1 Existing problem
 - 2.2 References
 - 2.3 Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
 - 3.1 Empathy Map Canvas
 - 3.2 Ideation & Brainstorming
4. **REQUIREMENT ANALYSIS**
 - 4.1 Functional requirement
 - 4.2 Non-Functional requirements
5. **PROJECT DESIGN**
 - 5.1 Data Flow Diagrams & User Stories
 - 5.2 Solution Architecture
6. **PROJECT PLANNING & SCHEDULING**
 - 6.1 Technical Architecture
 - 6.2 Sprint Planning & Estimation
 - 6.3 Sprint Delivery Schedule
7. **CODING & SOLUTIONING (Explain the features added in the project along with code)**
 - 7.1 Feature 1
 - 7.2 Feature 2
 - 7.3 Database Schema (if Applicable)
8. **PERFORMANCE TESTING**
 - 8.1 Performace Metrics
9. **RESULTS**
 - 9.1 Output Screenshots
10. **ADVANTAGES & DISADVANTAGES**
11. **CONCLUSION**
12. **FUTURE SCOPE**
13. **APPENDIX**
 - Source Code
 - GitHub & Project Demo Link

Introduction

Project Overview

The Snack Ordering App Development project aims to create a user-friendly and efficient mobile application that enables users to order a variety of snacks through delivery. The app is designed to cater to the needs of busy individuals who want a convenient way to satisfy their snack cravings without the hassle of visiting physical snack shops. This project involves the development of a comprehensive mobile application for Android platforms, along with a back-end system to manage orders, inventory, and user accounts.

Purpose

The Snack Ordering App Development project is driven by the goal of providing a user-friendly platform for snack enthusiasts to order their favourite snacks conveniently and offering snack vendors a tool to grow their businesses. This project aligns with the ever-increasing demand for efficient and technology-driven solutions in the food service industry, making it a valuable addition to the mobile app landscape.

Literature Survey

Existing Problem

Traditional snack ordering methods are often inconvenient and time-consuming for consumers, and snack vendors face challenges in expanding their customer reach and efficiently managing their operations.

References

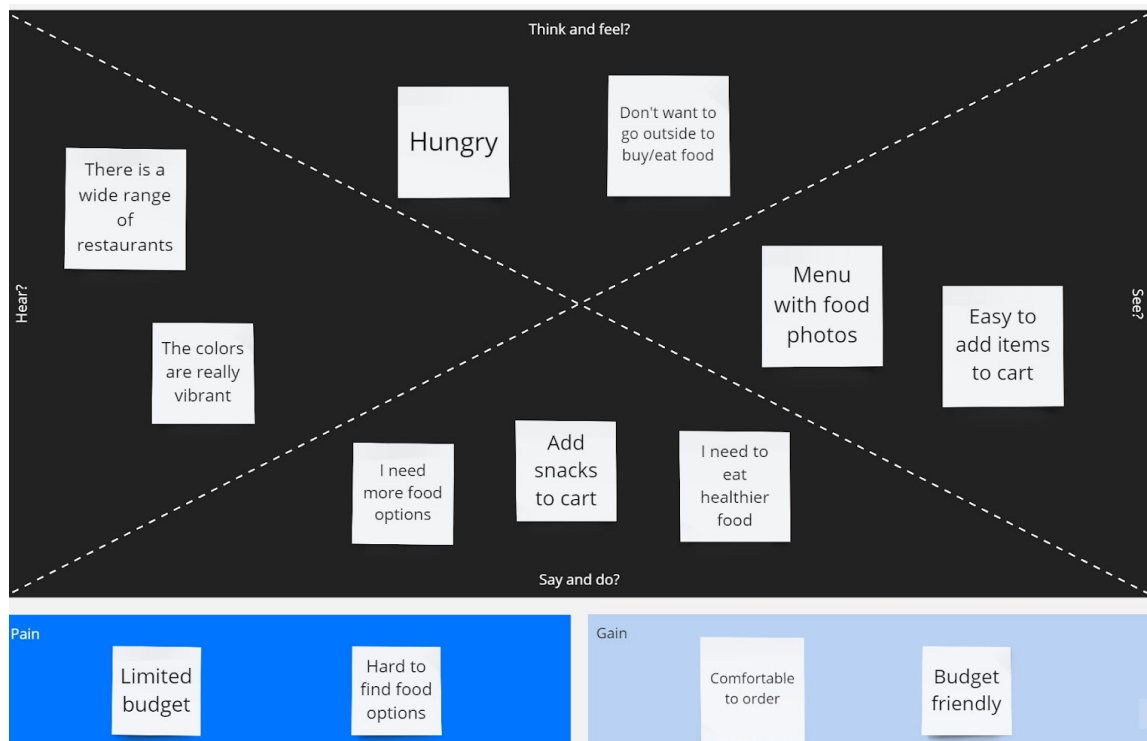
To address these existing problems and develop an effective Snack Ordering App, it is essential to study and draw inspiration from existing apps and relevant literature. Some references that may be useful for your project include existing food delivery apps, user feedback and reviews and academic research.

Problem Statement Definition

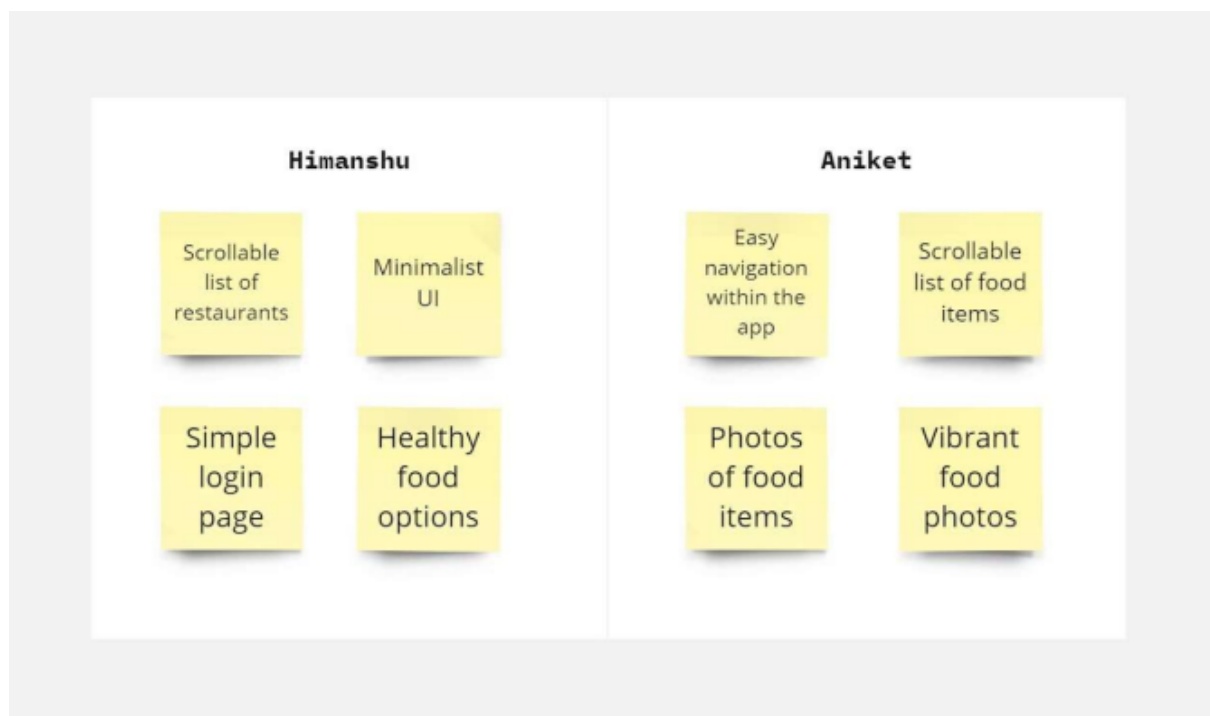
The Snack Ordering App Development project aims to address the existing problems and challenges in the snack ordering and delivery industry by developing a mobile application that offers a convenient, efficient, and personalised solution for consumers and snack vendors.

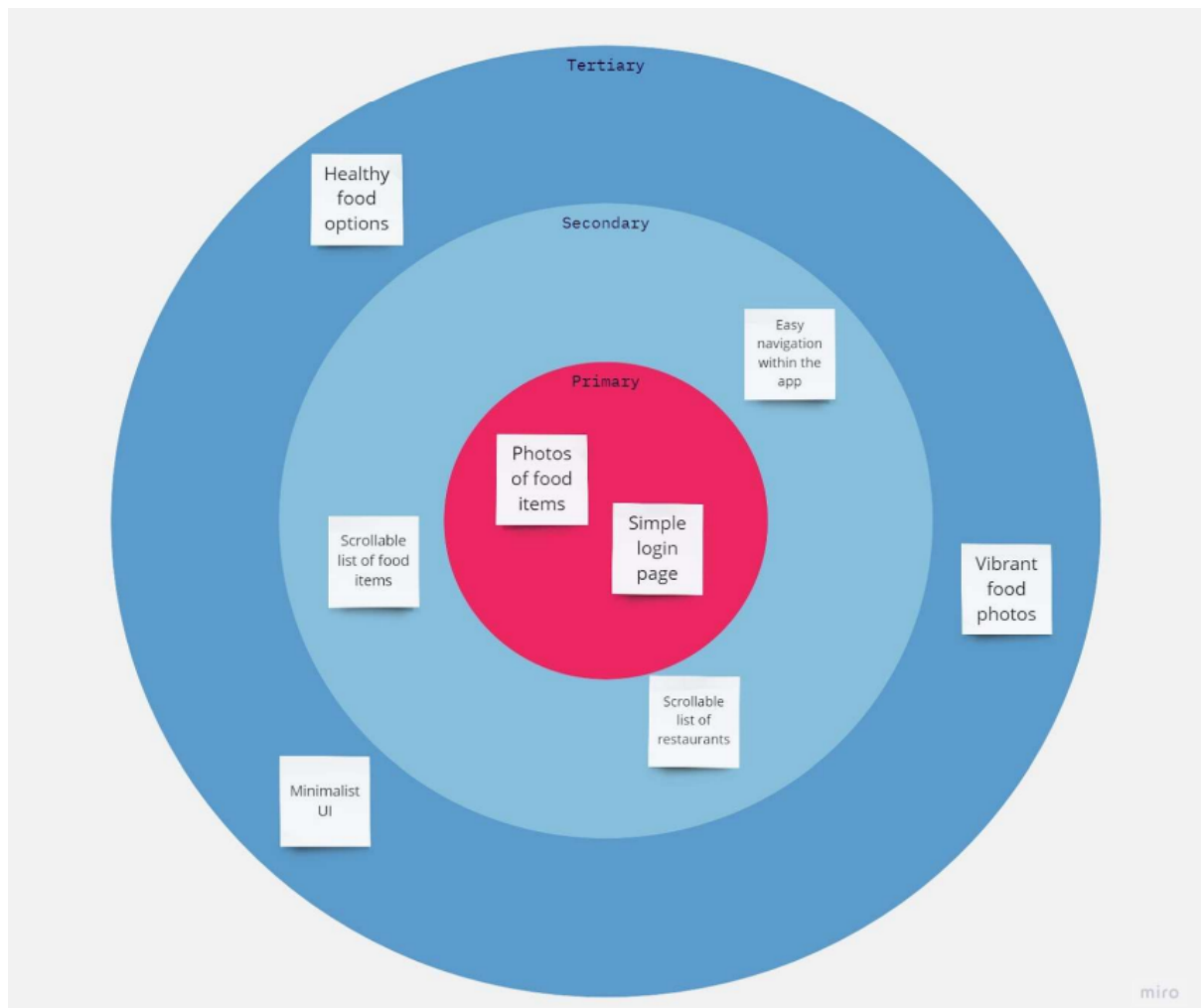
IDEATION & PROPOSED SOLUTION

Empathy Map Canvas



Ideation & Brainstorming





REQUIREMENT ANALYSIS

Functional requirement

User Registration and Authentication
Snack Menu
Order Placement
Rating and Reviews
Notifications

Non-Functional requirements

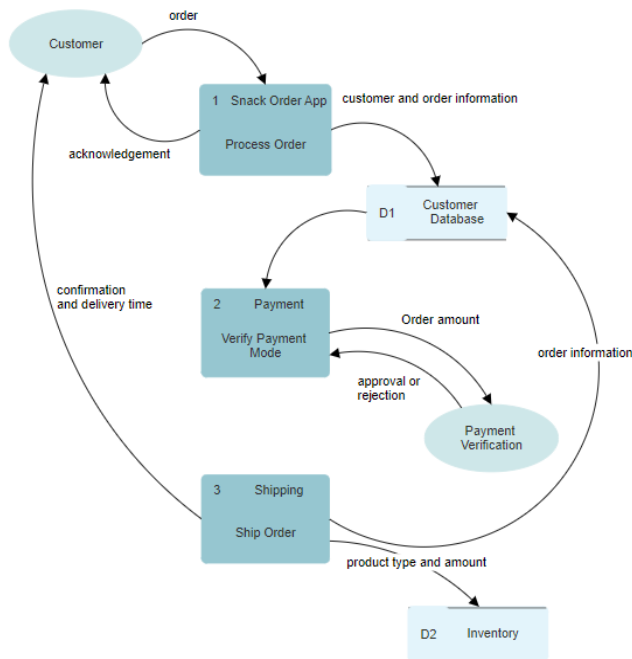
Performance
Security
User Experience (UX)
Documentation
Scalability

PROJECT DESIGN

Data Flow Diagrams & User Stories

Data Flow Diagram

Data Flow Diagram - Online Order System



User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer	Registration	USN-1	As a new user, I want to create an account to order snacks.	The user can register with a unique email address and password.	High	Sprint-1
		USN-2	*As a registered user, I want to manage my profile information.	The user can update their name, contact number, and delivery address in their profile.	Low	Sprint-1
	Snack Discovery	USN-1	As a hungry user, I want to explore available snacks	The user can browse a list of snacks, including images and descriptions.	High	Sprint-1
	Snack Ordering	USN-1	As a craving user, I want to add snacks to my shopping cart	The cart total is updated in real-time.	Medium	Sprint-2
		USN-2	As a user, I want to view and edit items	The user can view the contents of their	High	Sprint-2

			in my cart before placing an order.*	cart, including items and quantities.		
Vendor	Order Management	USN-1	As a snack vendor, I want to receive and manage incoming snack orders.	I can review order details, including the customer's name, delivery address, and the snack items.	High	Sprint-3
	Customer Interaction	USN-1	As a snack vendor, I want to manage customer reviews and ratings.	I can view customer reviews and ratings for my products and service.	Medium	Sprint-3

Solution Architecture

Frontend	The frontend of the programme will be in charge of presenting the user interface and handling user interactions. It will be implemented in java and xml and will be accessible from any device.
Backend	The app's backend will be in charge of processing orders and connecting with delivery services. It will be built with java and firebase.
Database	A database will be required for the app to hold data on users, snacks, orders, and delivery. This will be accomplished with the usage of Firebase.

PROJECT PLANNING & SCHEDULING

Technical Architecture

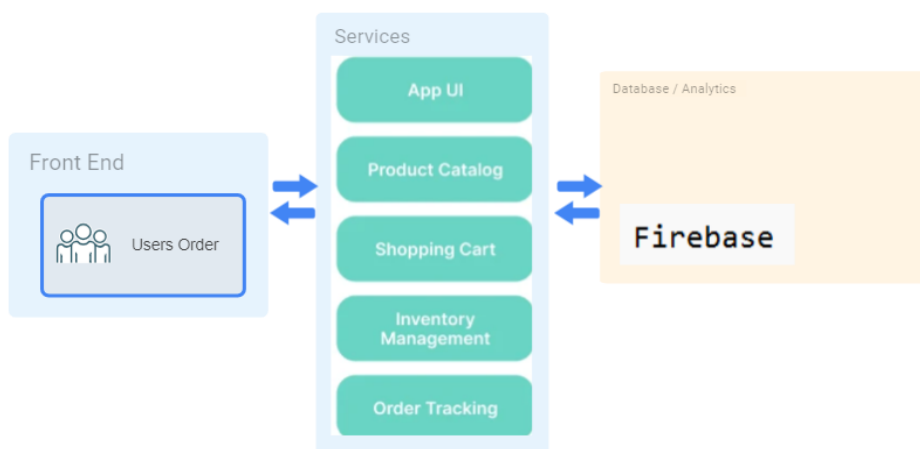


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	Client-Side (Mobile App)	The client-side is the mobile application that customers use to browse snacks, place orders, and track deliveries.	Java
2.	Server-Side	The server-side is responsible for processing customer orders, managing inventory, coordinating deliveries, and handling user authentication.	Java, Firebase
	Application Server	Contains business logic and communicates with the database.	Java, Firebase
	Authentication Service	Manages user registration, login, and authentication.	Java, Firebase
	Order Management	Processes orders, handles payments, and sends notifications to the customer.	Java, Firebase
	Delivery Management	Assigns delivery drivers.	Java, Firebase
3.	Database	The database stores information about snacks, users, orders, and delivery drivers.	Firebase

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Scalability	The ability to handle a large number of concurrent users and orders.	Firebase
2.	Reliability	The ability to be available and perform consistently even under high load or in the event of failures.	Firebase
3.	Performance	The ability to respond to user requests quickly and efficiently.	Firebase
4.	Security	The ability to protect user data and prevent unauthorised access.	Firebase

Sprint Planning & Estimation

Product Backlog, Sprint Schedule, and Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Member
Sprint-1	Registration	USN-1	As a new user, I want to create an account to order snacks.	2	High	Aniket
Sprint-1		USN-2	As a registered user, I want to manage my profile information.	1	Low	Himanshu
Sprint-1	Snack Discovery	USN-1	As a hungry user, I want to explore available snacks	1	High	Himanshu
Sprint-2	Snack Ordering	USN-1	As a craving user, I want to add snacks to my shopping cart	2	Medium	Aniket
Sprint-2		USN-2	As a user, I want to view and edit items in my cart before placing an order.	1	High	Himanshu, Aniket
Sprint-3	Order Management	USN-1	As a snack vendor, I want to receive and manage incoming snack orders.	1	High	Aniket
Sprint-3	Customer Interaction	USN-1	As a snack vendor, I want to manage customer reviews and ratings.	2	Medium	Himanshu

Sprint Delivery Schedule

Project Tracker, Velocity & Burndown Chart:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	10	5 Days	3 Oct 2023	7 Oct 2023	10	7 Oct 2023
Sprint-2	10	5 Days	8 Oct 2023	12 Oct 2023	10	12 Oct 2023
Sprint-3	10	5 Days	13 Oct 2023	17 Oct 2023	8	18 Oct 2023

CODING & SOLUTIONING

Feature 1

Signup/Login

```
package my.snacksquad.app;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.app.ProgressDialog;
import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.android.material.textfield.TextInputLayout;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;

public class Login extends AppCompatActivity {

    TextInputLayout email,pass;
    Button Signin,signupp;
    TextView Forgotpassword , signup;
    FirebaseAuth Fauth;
    String emailid,pwd;

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

        try{

            email = (TextInputLayout)findViewById(R.id.email);
            pass = (TextInputLayout)findViewById(R.id.Lpassword);
            Signin = (Button)findViewById(R.id.button4);
            signupp = (Button) findViewById(R.id.newbtn);
            Forgotpassword = (TextView)findViewById(R.id.forgotpass);

            Fauth = FirebaseAuth.getInstance();

            Signin.setOnClickListener(new View.OnClickListener() {
```

```

@Override
public void onClick(View v) {

    emailid = email.getText().getText().toString().trim();
    pwd = pass.getText().getText().toString().trim();

    if(isValid()){

        final ProgressDialog mDialog = new ProgressDialog(Login.this);
        mDialog.setCanceledOnTouchOutside(false);
        mDialog.setCancelable(false);
        mDialog.setMessage("Sign In Please Wait.....");
        mDialog.show();

Fauth.signInWithEmailAndPassword(emailid,pwd).addOnCompleteListener(new
OnCompleteListener<AuthResult>() {
    @Override
    public void onComplete(@NonNull Task<AuthResult> task) {

        if(task.isSuccessful()){
            mDialog.dismiss();

            if(Fauth.getCurrentUser().isEmailVerified()){
                mDialog.dismiss();
                Toast.makeText(Login.this, "Congratulation! You Have Successfully
Login", Toast.LENGTH_SHORT).show();
                Intent Z = new Intent(Login.this,FoodPanel.class);
                startActivity(Z);
                finish();

            }else{
                Extra.ShowAlert(Login.this,"Verification Failed","You Have Not
Verified Your Email");

            }
        }else{
            mDialog.dismiss();
            Extra.ShowAlert(Login.this,"Error",task.getException().getMessage());
        }
    }
});
    }
});
signupp.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

```

```

        startActivity(new Intent(Login.this,signuppage.class));
        finish();
    }
});
Forgotpassword.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        startActivity(new Intent(Login.this,forgotpassword.class));
        finish();
    }
});
}catch (Exception e){
    Toast.makeText(this,e.getMessage(),Toast.LENGTH_LONG).show();
}

}
String emailpattern = "[a-zA-Z0-9._-]+@[a-z]+\\.[a-z]+";

public boolean isValid(){

    email.setErrorEnabled(false);
    email.setError("");
    pass.setErrorEnabled(false);
    pass.setError("");

    boolean invalid=false,invalidemail=false,invalidpassword=false;
    if(TextUtils.isEmpty(emailid)){
        email.setErrorEnabled(true);
        email.setError("Email is required");
    }else{
        if(emailid.matches(emailpattern)){
            invalidemail=true;
        }else{
            email.setErrorEnabled(true);
            email.setError("Invalid Email Address");
        }
    }
    if(TextUtils.isEmpty(pwd)){
        pass.setErrorEnabled(true);
        pass.setError("Password is Required");
    }else{
        invalidpassword=true;
    }
    invalid=(invalidemail && invalidpassword)?true:false;
    return invalid;
}

```

```

}

package my.snacksquad.app;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.app.AlertDialog;
import android.app.ProgressDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.Spinner;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.android.material.textfield.TextInputLayout;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.hbb20.CountryCodePicker;
import java.util.HashMap;
public class signuptime extends AppCompatActivity {
    TextInputLayout FName,Lname,Email,Pass,cpass,mobilen,house,area,pincode, city;
    Spinner Statespin;
    Button signup, Email, Phone;
    CountryCodePicker Cpp;
    FirebaseAuth FAuth;
    DatabaseReference databaseReference;
    FirebaseDatabase firebaseDatabase;
    String
fname,lname,emailid,password,confpassword,mobile,house,Area,Pincode,state,City;
    String role="Customer";

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

        FName = (TextInputLayout)findViewById(R.id.Firstname);
        Lname = (TextInputLayout)findViewById(R.id.Lastname);
        Email = (TextInputLayout)findViewById(R.id.Email);
        Pass = (TextInputLayout)findViewById(R.id.Pwd);
        cpass = (TextInputLayout)findViewById(R.id.Cpass);

```

```
mobilenno = (TextInputLayout)findViewById(R.id.Mobileno);
houseno = (TextInputLayout)findViewById(R.id.houseNo);
pincode = (TextInputLayout)findViewById(R.id.Pincode);
Statespin = (Spinner) findViewById(R.id.State);
city = (TextInputLayout) findViewById(R.id.City);
area = (TextInputLayout)findViewById(R.id.Area);
```

```
signup = (Button)findViewById(R.id.Signup);
```

```
Cpp = (CountryCodePicker)findViewById(R.id.CountryCode);
```

```
Statespin.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
    @Override
    public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {

        Object value = parent.getItemAtPosition(position);
        state = value.toString().trim();

    }

    @Override
    public void onNothingSelected(AdapterView<?> parent) {

    }
});
```

```
databaseReference = firebaseDatabase.getInstance().getReference("Chef");
FAuth = FirebaseAuth.getInstance();
```

```
signup.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

        fname = Fname.getText().getText().toString().trim();
        lname = Lname.getText().getText().toString().trim();
        emailid = Email.getText().getText().toString().trim();
        mobile = mobilenno.getText().getText().toString().trim();
        password = Pass.getText().getText().toString().trim();
        confpassword = cpass.getText().getText().toString().trim();
        Area = area.getText().getText().toString().trim();
        house = houseno.getText().getText().toString().trim();
        Pincode = pincode.getText().getText().toString().trim();

        if (isValid()){
            final ProgressDialog mDialog = new ProgressDialog(signuppage.this);
            mDialog.setCancelable(false);
            mDialog.setCanceledOnTouchOutside(false);
```

```
mDialog.setMessage("Registration in progress please wait.....");
mDialog.show();
```

```
FAuth.createUserWithEmailAndPassword(emailid,password).addOnCompleteListener(new
OnCompleteListener<AuthResult>() {
```

```
    @Override
```

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

```
        if (task.isSuccessful()){
```

```
            String useridd = FirebaseAuth.getInstance().getCurrentUser().getUid();
```

```
            databaseReference =
```

```
            FirebaseDatabase.getInstance().getReference("User").child(useridd);
```

```
            final HashMap<String , String> hashMap = new HashMap<>();
```

```
            hashMap.put("Role",role);
```

```
FAuth.getCurrentUser().sendEmailVerification().addOnCompleteListener(new
OnCompleteListener<Void>() {
```

```
    @Override
```

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

```
        if(task.isSuccessful()){
```

```
            mDialog.dismiss();
```

```
            AlertDialog.Builder builder = new
```

```
AlertDialog.Builder(signuppage.this);
```

```
            builder.setMessage("You Have Registered! Make Sure To Verify
```

```
Your Email");
```

```
            builder.setCancelable(false);
```

```
            builder.setPositiveButton("Ok", new
```

```
DialogInterface.OnClickListener() {
```

```
    @Override
```

```
    public void onClick(DialogInterface dialog, int which) {
```

```
        Intent c = new Intent(signuppage.this,FoodPanel.class);
```

```
        startActivity(c);
```

```
        finish();
```

```
    }
```

```
});
```

```
AlertDialog Alert = builder.create();
```

```
Alert.show();
```

```
}else{
```

```
    mDialog.dismiss();
```

```
Extra.ShowAlert(signuppage.this,"Error",task.getException().getMessage());
```

```
}
```

```

        }
    });
}
}
});
}
//
}
});
}

```

```

String emailpattern = "[a-zA-Z0-9._-]+@[a-z]+\\.[a-z]+";
public boolean isValid(){
    Email.setErrorEnabled(false);
    Email.setError("");
    Fname.setErrorEnabled(false);
    Fname.setError("");
    Lname.setErrorEnabled(false);
    Lname.setError("");
    Pass.setErrorEnabled(false);
    Pass.setError("");
    mobileno.setErrorEnabled(false);
    mobileno.setError("");
    cpass.setErrorEnabled(false);
    cpass.setError("");
    area.setErrorEnabled(false);
    area.setError("");
    houseno.setErrorEnabled(false);
    houseno.setError("");
    pincode.setErrorEnabled(false);
    pincode.setError("");

    boolean
isValid=false,isValidhouseno=false,isValidlname=false,isValidname=false,isValidemail=false,i
sValidpassword=false,isValidconfpassword=false,isValidmobilenumber=false,isValidarea=false,i
sValidpincode=false;
    if(TextUtils.isEmpty(fname)){
        Fname.setErrorEnabled(true);
        Fname.setError("Enter First Name");
    }else{
        isValidname = true;
    }
    if(TextUtils.isEmpty(lname)){
        Lname.setErrorEnabled(true);
        Lname.setError("Enter Last Name");
    }
}

```

```

}else{
    isValidName = true;
}
if(TextUtils.isEmpty(emailid)){
    Email.setErrorEnabled(true);
    Email.setError("Email Is Required");
}else{
    if(emailid.matches(emailpattern)){
        isValidEmail = true;
    }else{
        Email.setErrorEnabled(true);
        Email.setError("Enter a Valid Email Id");
    }
}
if(TextUtils.isEmpty(password)){
    Pass.setErrorEnabled(true);
    Pass.setError("Enter Password");
}else{
    if(password.length()<8){
        Pass.setErrorEnabled(true);
        Pass.setError("Password is Weak");
    }else{
        isValidPassword = true;
    }
}
if(TextUtils.isEmpty(confpassword)){
    cpass.setErrorEnabled(true);
    cpass.setError("Enter Password Again");
}else{
    if(!password.equals(confpassword)){
        cpass.setErrorEnabled(true);
        cpass.setError("Password Doesn't Match");
    }else{
        isValidConfpassword = true;
    }
}
if(TextUtils.isEmpty(mobile)){
    mobileno.setErrorEnabled(true);
    mobileno.setError("Mobile Number Is Required");
}else{
    if(mobile.length()<10){
        mobileno.setErrorEnabled(true);
        mobileno.setError("Invalid Mobile Number");
    }else{
        isValidMobilenum = true;
    }
}
if(TextUtils.isEmpty(Area)){

```



```

        area.setErrorEnabled(true);
        area.setError("Area Is Required");
    }else{
        isValidarea = true;
    }
    if(TextUtils.isEmpty(Pincode)){
        pincode.setErrorEnabled(true);
        pincode.setError("Please Enter Pincode");
    }else{
        isValidpincode = true;
    }
    if(TextUtils.isEmpty(house)){
        houseno.setErrorEnabled(true);
        houseno.setError("Fields Can't Be Empty");
    }else{
        isValidhouseno = true;
    }
}

isValid = (isValidarea && isValidconfpassword && isValidpassword && isValidpincode &&
isValidemail && isValidmobilenumber && isValidname && isValidhouseno && isValidlname) ?
true : false;
return isValid;

}
}

```

Feature 2

Food Menu with Rating

```

package my.snacksquad.app;

import android.os.Parcel;
import android.os.Parcelable;

public class dishh implements Parcelable {
    private String name;
    private String description;
    private String rating;
    private String price;
    private int id;
    private int quantity;
    protected dishh(Parcel in) {
        id = in.readInt();
        name = in.readString();
        description = in.readString();
        rating = in.readString();
    }
}

```

```

        price = in.readString();
        quantity = in.readInt();
    }
    public static final Creator<dishh> CREATOR = new Creator<dishh>() {
        @Override
        public dishh createFromParcel(Parcel in) {
            return new dishh(in);
        }

        @Override
        public dishh[] newArray(int size) {
            return new dishh[size];
        }
    };

    @Override
    public int describeContents() {
        return 0;
    }

    @Override
    public void writeToParcel(Parcel dest, int flags) {
        dest.writeInt(id);
        dest.writeString(name);
        dest.writeString(description);
        dest.writeString(rating);
        dest.writeString(price);
        dest.writeInt(quantity);
    }

    public dishh(int id,String name, String description, String rating, String price) {
        this.id = id;

        this.name = name;
        this.description = description;
        this.rating = rating;
        this.price = price;
    }

    public int getId() {
        return id;
    }

    public String getName() {
        return name;
    }
}

```

```

    public String getDescription() {
        return description;
    }

    public String getRating() {
        return rating;
    }

    public String getPrice() {
        return price;
    }
    public int getQuantity() {
        return quantity;
    }

    public void setQuantity(int quantity) {
        this.quantity = quantity;
    }
}

```

Feature 3

Cart

```

package my.snacksquad.app;

import java.util.ArrayList;
import java.util.List;

import androidx.lifecycle.LiveData;
import androidx.lifecycle.MutableLiveData;

public class Cart {
    private String name;
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }

    public String getPrice() {
        return price;
    }

    public void setPrice(String price) {
        this.price = price;
    }
}

```

```

public int getId() {
    return id;
}

public void setId(int id) {
    this.id = id;
}

private String price;
private int id;
private static Cart instance;
private List<dishh> cartItems;
private MutableLiveData<List<dishh>> cartLiveData = new MutableLiveData<>();
private Cart() {
    cartItems = new ArrayList<>();
}

public static Cart getInstance() {
    if (instance == null) {
        instance = new Cart();
    }
    return instance;
}

public void addToCart(dishh item) {
    // Check if the item is already in the cart
    boolean isItemInCart = false;

    for (dishh cartItem : cartItems) {
        if (cartItem.getId() == item.getId()) {
            isItemInCart = true;
            break;
        }
    }

    if (!isItemInCart) {
        item.setQuantity(1);
        cartItems.add(item);
    }

    notifyCartChanged();
}

public void removeFromCart(dishh item) {
    cartItems.remove(item);
    notifyCartChanged();
}

public double getTotalPrice() {

```

```

        double totalPrice = 0.0;

        for (dishh item : cartItems) {
            try {
                double itemPrice = Double.parseDouble(item.getPrice());
                totalPrice += itemPrice * item.getQuantity();
            } catch (NumberFormatException e) {
            }
        }

        return totalPrice;
    }

    public LiveData<List<dishh>> getCartLiveData() {
        return cartLiveData;
    }

    public List<dishh> getCartItems() {
        return cartItems;
    }

    private void notifyCartChanged() {
        // Notify LiveData observers of changes
        cartLiveData.setValue(new ArrayList<>(cartItems));
    }
}

```

Feature 4

Profile

```

package my.snacksquad.app;

import android.content.Intent;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import androidx.fragment.app.Fragment;
import android.widget.Button;
import android.widget.TextView;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
public class ProfileFragment extends Fragment {
    private TextView emailTextView;
    private Button logoutButton;

    public ProfileFragment() {

```

```

    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle
savedInstanceState) {
        View view = inflater.inflate(R.layout.fragment_profile, container, false);
        emailTextView = view.findViewById(R.id.emailTextView);
        logoutButton = view.findViewById(R.id.logoutButton);
        FirebaseAuth user = FirebaseAuth.getInstance().getCurrentUser();

        if (user != null) {
            String email = user.getEmail();
            emailTextView.setText(email);
        }
        logoutButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                FirebaseAuth.getInstance().signOut();

                Intent mainMenuIntent = new Intent(getActivity(), Mainmenu.class);
                startActivity(mainMenuIntent);
                getActivity().finish();
            }
        });

        return view;
    }

}

```

Feature 5

Food Panel

```

package my.snacksquad.app;

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 my.snacksquad.app.databinding.ActivityFoodPanelBinding;

public class FoodPanel extends AppCompatActivity {
    ActivityFoodPanelBinding binding;
}

```

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    binding = ActivityFoodPanelBinding.inflate(getLayoutInflater());
    setContentView(binding.getRoot());
    replaceFragment(new HomeFragment());
    binding.bottomNavigationView.setBackground(null);

    binding.bottomNavigationView.setOnItemSelectedListener(item -> {
        if (item.getItemId() == R.id.home) {
            replaceFragment(new HomeFragment());
        } else if (item.getItemId() == R.id.cart) {
            replaceFragment(new CartFragment());
        } else if (item.getItemId() == R.id.profile) {
            replaceFragment(new ProfileFragment());
        }
        return true;
    });
}

private void replaceFragment(Fragment fragment){
    FragmentManager fragmentManager = getSupportFragmentManager();
    FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();
    fragmentTransaction.replace(R.id.frame_layout, fragment);
    fragmentTransaction.commit();
}
}

```

Feature 6

Forgot Password

```

package my.snacksquad.app;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.FirebaseAuth;

public class forgotpassword extends AppCompatActivity {

```

```

private EditText emailEditText;
private Button resetPasswordButton;
private FirebaseAuth firebaseAuth;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_forgotpassword);
    emailEditText = findViewById(R.id.emailEditText);
    resetPasswordButton = findViewById(R.id.resetPasswordButton);

    firebaseAuth = FirebaseAuth.getInstance();

    resetPasswordButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            String email = emailEditText.getText().toString().trim();

            if (email.isEmpty()) {
                emailEditText.setError("Please enter your email");
                return;
            }

            // Send a password reset email to the specified email address
            firebaseAuth.sendPasswordResetEmail(email)
                .addOnCompleteListener(new OnCompleteListener<Void>() {
                    @Override
                    public void onComplete(@NonNull Task<Void> task) {
                        if (task.isSuccessful()) {
                            // Password reset email sent successfully
                            Toast.makeText(forgotpassword.this, "Password reset email sent.
Check your inbox.", Toast.LENGTH_SHORT).show();
                            finish();
                        } else {
                            // Password reset email failed to send
                            Toast.makeText(forgotpassword.this, "Failed to send password reset
email.", Toast.LENGTH_SHORT).show();
                        }
                    }
                });
        }
    });
}

```

Database Schema

Used Firebase

PERFORMANCE TESTING

Performance Metrics

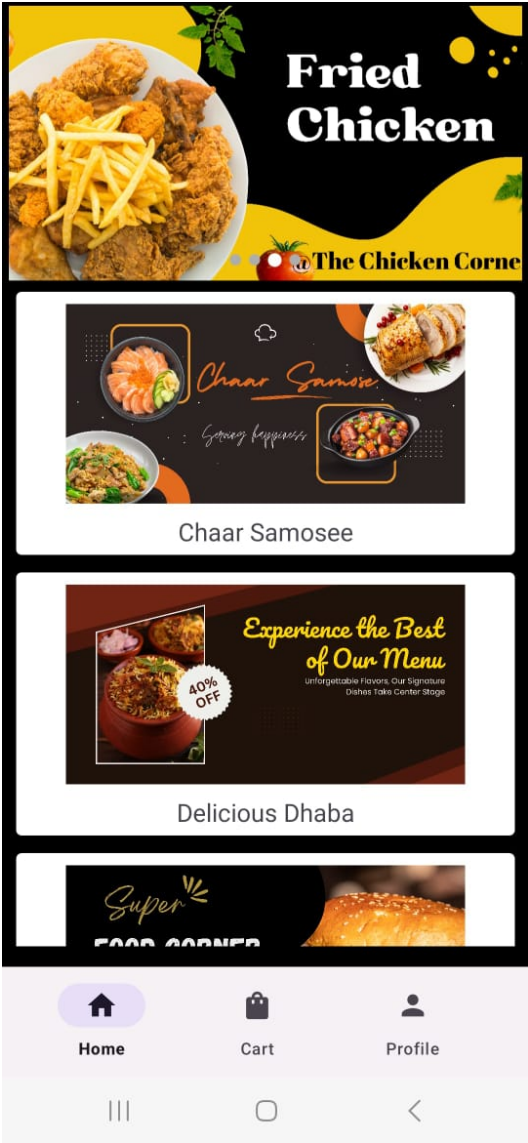
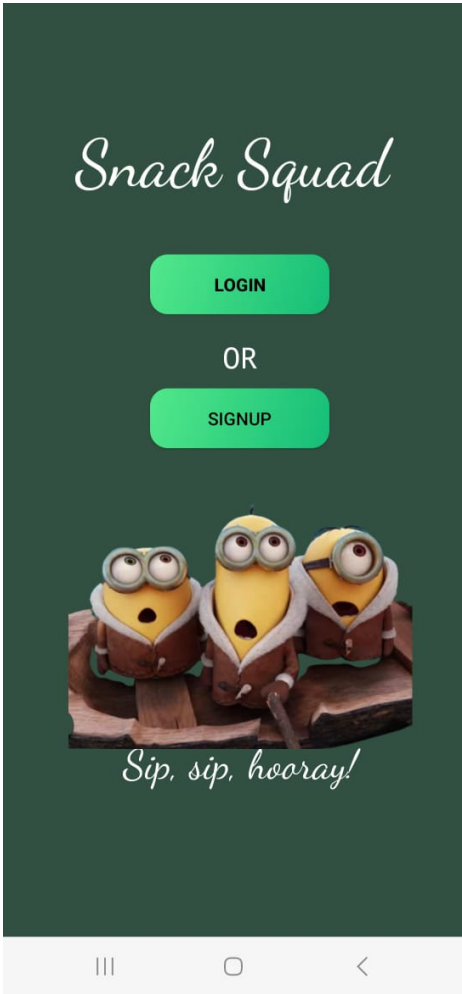
Cross Platform Compatibility: It works only for android.


Load Bearing Compatibility: Average

Reliability Testing: Reliable

RESULTS

Output Screenshots






Biryani
Description

★ 5.0

₹ 349

Remove

- 1 +




Chicken Grilled
Description

★ 5.0

₹ 349

Remove

- 1 +



Pizza
Description

★ 5.0

₹ 649

Remove

- 1 +

Total Price: 1347.0

Place Order


Your Orders

Home

Cart

Profile

Welcome to
Chaar Samose




Biryani
Description

★ 5

₹ 349

Add to Cart

- 2 +




Chicken Grilled
Description

★ 4.9

₹ 349

Add to Cart

- 1 +




Pizza
Description

★ 4

₹ 649

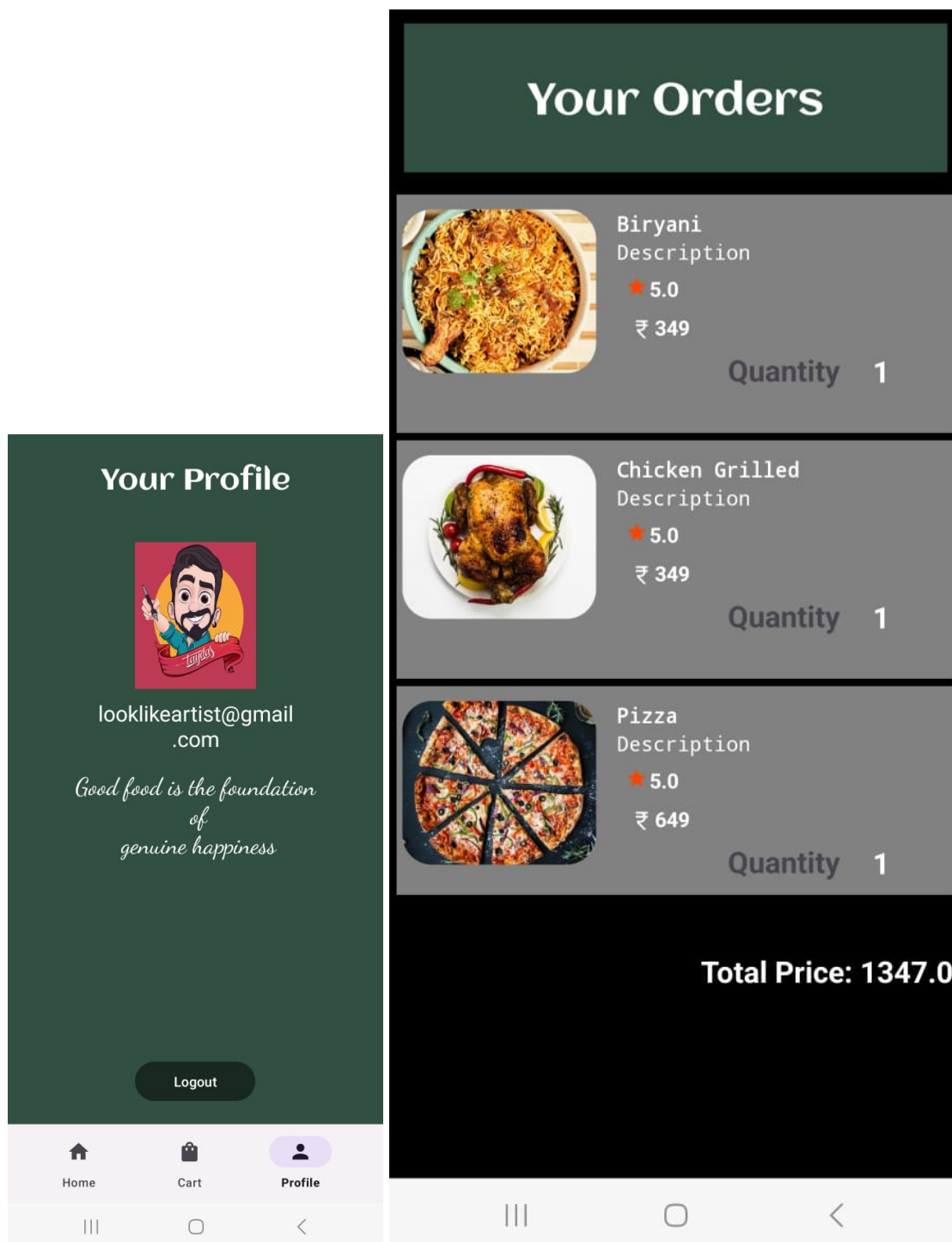
Add to Cart

- 3 +



Burger
Description

★ 5



ADVANTAGES & DISADVANTAGES

Advantages

- Convenience
- Accessibility
- Efficiency
- User Reviews
- Vendor Opportunities

Disadvantages

Tech Dependency

Delivery Delays

Delivery Costs

Limited Physical Experience

Competitive Market

CONCLUSION

In conclusion, the Snack Ordering App Development project represents an exciting and innovative solution to address the existing challenges in the snack ordering and delivery industry. By carefully considering the project overview, purpose, existing problems, references, and problem statement definition, we have laid a strong foundation for the development of a user-friendly and efficient app.

The Snack Ordering App project presents a promising opportunity to transform the way consumers access and enjoy their favourite snacks, while also providing a platform for snack vendors to thrive in an increasingly digital landscape. With a strong focus on user experience, security, and performance, the app has the potential to become a valuable addition to the snack ordering market, offering convenience and choice to snack enthusiasts everywhere.

FUTURE SCOPE

The future scope of the Snack Ordering App is promising, with opportunities for geographical expansion, international market entry, integration with emerging technologies like drones and AI-driven chatbots, smart kitchen partnerships, sustainability initiatives, offline access, IoT device integration, innovative marketing strategies, diversified snack offerings, data-driven personalization, enhanced user support, partnerships with food festivals and events, user and vendor analytics, blockchain for food safety, and adoption of voice commerce. Adapting to evolving consumer preferences and technological advancements will be crucial in maintaining the app's competitiveness and ensuring sustained growth in the food delivery and mobile app landscape.

APPENDIX

Source Code

<https://github.com/dwivedianiket/SnackSquad/tree/master>

GitHub & Project Demo Link

<https://github.com/dwivedianiket/SnackSquad/tree/master>

https://drive.google.com/file/d/17UfkYpFEq82APS9W9McBIN7jB_yXhalt/view?usp=sharing