Project Report

Android Application Development

SMARTBRIDGE
Let's Bridge the Gap

Smart Internz

A Customizable Snack Ordering and Delivery App

TEAM ID: 591092

LIST OF CONTENTS

Source Code:
https://drive.google.com/drive/folders/1KUe8pLVjAZjIeSVFWLSJxWD1gt9Nmq0k?usp=sharing

GitHub & Project Demo Link:
https://github.com/smartinternz02/SI-GuidedProject-587226-1697082986

INTRODUCTION

1.1Project Overview:

Creating a snacks ordering app in Kotlin with Android Studio involves several key steps. Begin by setting up a new Kotlin project in Android Studio, defining the project structure, and configuring dependencies. Users can typically search for nearby restaurants, explore menus, view food images and descriptions, and even read reviews from other customers. Once an order is placed, users can track the progress of their delivery and receive updates on estimated arrival times. Snack ordering apps often offer features like secure payment options, order history tracking, and the ability to save favorite orders or restaurants for future reference. Finally, prepare the app for deployment by configuring build settings and generating a signed APK or bundle for distribution on app stores like the Google Play Store. Throughout the development process, adhere to best practices for Android development, ensuring code modularity, responsive user interactions, and optimal performance.

1.2 Purpose:

The purpose of a snack ordering app is to provide a convenient and efficient way for users to order snacks from a variety of restaurants or food establishments. These apps aim to simplify the process of browsing menus, customizing orders, and placing them for delivery or pickup. By using a snack ordering app, users can save time and effort by avoiding the need to physically visit a restaurant or make phone calls to place their orders. The app also allows for easy payment options, order tracking, and the ability to explore different food options and reviews. Overall, the snack ordering app is designed to cater to users' preferences, providing a variety of snack options and fostering a sense of community engagement in the realm of mobile snack consumption.

2. LITERATURE SURVEY

2.1 Existing Problem:

Online snack ordering app is popular, but there's a gap for a dedicated snack ordering platform. Current solutions don't meet the convenience and efficiency users want for quick snacks. Issues include limited snack variety and user-friendly features. Our app will bridge this gap with tailored solutions, prioritizing preferences, diverse snack options, and a seamless, secure interface.

2.2 References :

Shah, N., Shah, H., Sheth, S., Chauhan, R., & Desai, C. (2021, May). Local Food Delivery System. In Proceedings of the 4th International Conference on Advances in Science & Technology (ICAST2021).

2.3 Problem Statement Definition :

It lacks efficiency, making it difficult for customers to quickly place their orders. Additionally, the limited snack options result in a lack of variety, restricting customers' choices based on their preferences and dietary requirements. It's also challenging for customers with specific dietary needs to make informed choices since the platform doesn't provide sufficient dietary information about the snacks. These are definitely areas that could use improvement to enhance the overall snack ordering experience!

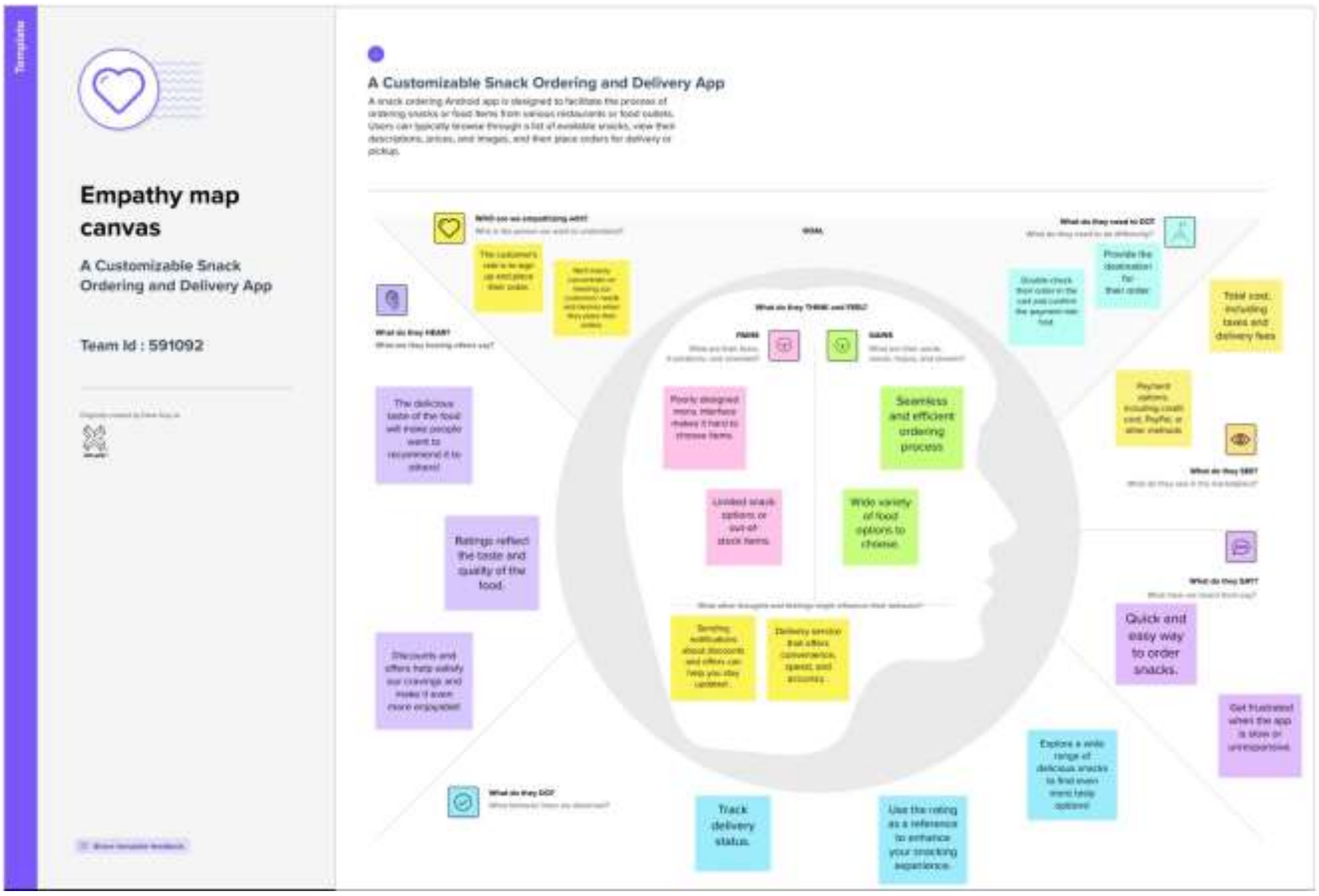
3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas :

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes.
It is a useful tool to helps teams better understand their users.
Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.

Empathy Map :

A


Template
Empathy map canvas
A Customizable Snack Ordering and Delivery App
Team Id : 591092
A Customizable Snack Ordering and Delivery App
A snack ordering Android app is designed to facilitate the process of ordering snacks or food items from various restaurants or food outlets. Users can typically browse through a list of available snacks, view their descriptions, prices, and images, and then place orders for delivery or pickup.
GOAL
What do they HEAR?
What do they THINK and FEEL?
PAINS
GAINS
What do they need to DO?
What do they SEE?
What do they SAY?
What do they DO?
The delicious taste of the food will make people want to recommend it to others!
Ratings reflect the taste and quality of the food.
Discounts and offers help satisfy our cravings and make it even more enjoyable!
Poorly designed menu interface makes it hard to choose items.
Limited snack options or out-of-stock items.
Seamless and efficient ordering process
Wide variety of food options to choose.
Sending notifications about discounts and offers can help you stay updated.
Delivery service that offers convenience, speed, and accuracy.
Provide the destination for their order
Total cost, including taxes and delivery fees
Payment options, including credit card, PayPal, or other methods
Quick and easy way to order snacks.
Get frustrated when the app is slow or unresponsive.
Track delivery status.
Use the rating as a reference to enhance your snacking experience.
Explore a wide range of delicious snacks to find even more tasty options!

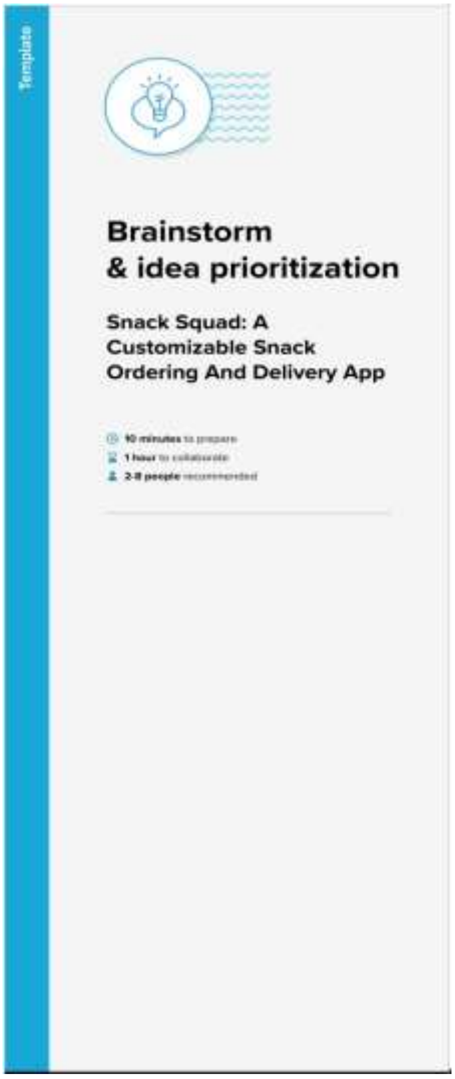
Reference:
https://app.mural.co/t/snacksquadandroidproject6951/m/snacksquadandroidproject6951/1697014175711/bfc827da2395f82b40aee062cc452589083bb4e2?sender=uef199e6ce421cd7f6aec4839

3.2 Brainstorm & Idea Prioritization Template:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

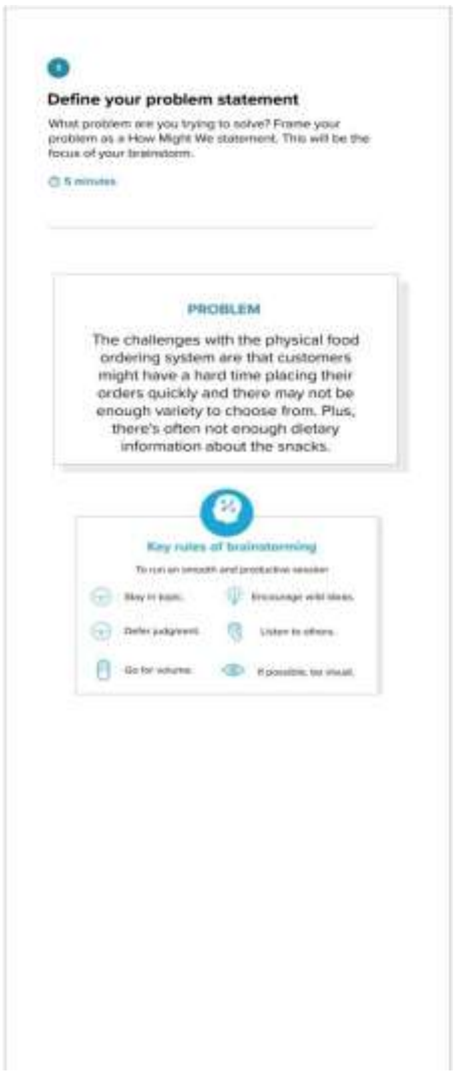
Step-1: Team Gathering, Collaboration and Select the Problem Statement



Brainstorm & idea prioritization

Snack Squad: A Customizable Snack Ordering And Delivery App

10 minutes to prepare
1 hour to collaborate
2-8 people recommended



Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

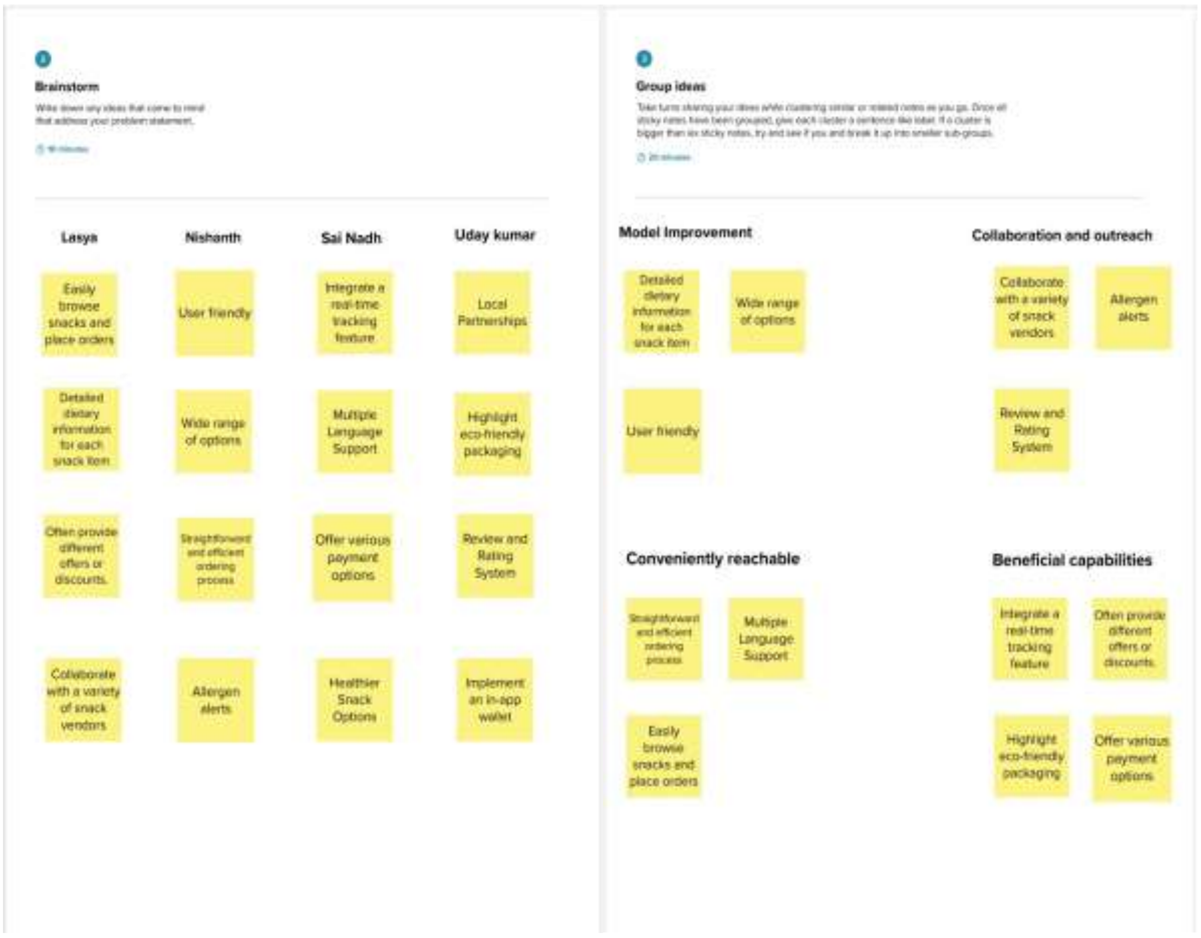
The challenges with the physical food ordering system are that customers might have a hard time placing their orders quickly and there may not be enough variety to choose from. Plus, there's often not enough dietary information about the snacks.

Key rules of brainstorming

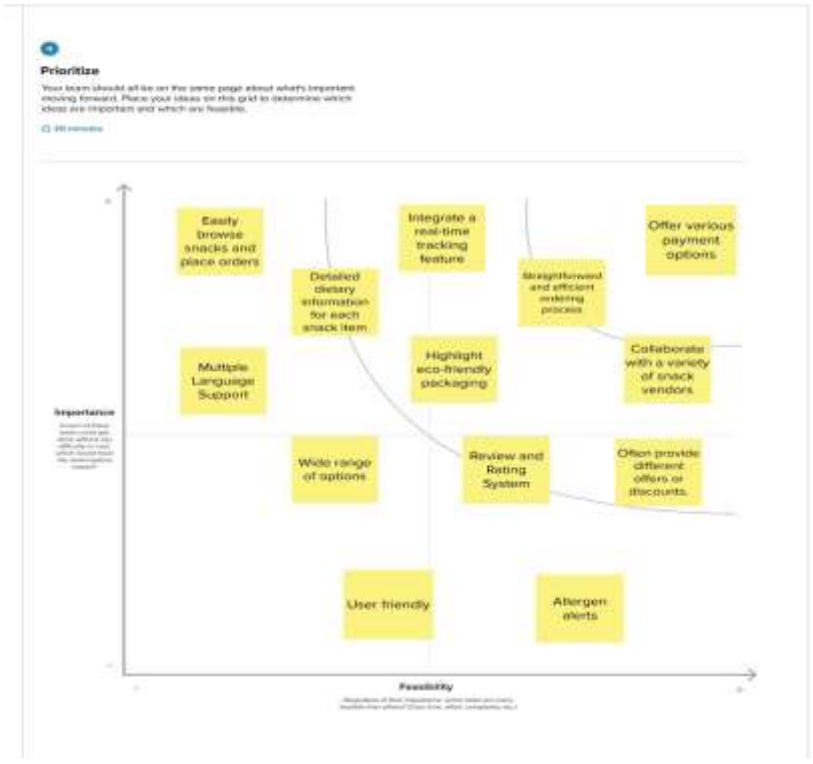
To run an smooth and productive session

Stay in topic.	Encourage wild ideas.
Defer judgment.	Listen to others.
Go for volume.	If possible, be visual.

Step-2: Brainstorm, Idea Listing and Grouping


Brainstorm
Lasya
Nishanth
Sai Nadh
Uday kumar
Easily browse snacks and place orders
User friendly
Integrate a real-time tracking feature
Local Partnerships
Detailed dietary information for each snack item
Wide range of options
Multiple Language Support
Highlight eco-friendly packaging
Often provide different offers or discounts.
Straightforward and efficient ordering process
Offer various payment options
Review and Rating System
Collaborate with a variety of snack vendors
Allergen alerts
Healthier Snack Options
Implement an in-app wallet
Group ideas
Model Improvement
Detailed dietary information for each snack item
Wide range of options
User friendly
Collaboration and outreach
Collaborate with a variety of snack vendors
Allergen alerts
Review and Rating System
Conveniently reachable
Straightforward and efficient ordering process
Multiple Language Support
Easily browse snacks and place orders
Beneficial capabilities
Integrate a real-time tracking feature
Often provide different offers or discounts.
Highlight eco-friendly packaging
Offer various payment options

Step-3: Idea Prioritization



Reference Link:
https://app.mural.co/t/snacksquadandroidproject6951/m/snacksquadandroidproject6951/1697266749390/495354d055996afb22ad70387440c9eb29ce36db?sender=uef199e6ce421cd7f6ae c4839

4. REQUIREMENT ANALYSIS

4.1 Functional requirement :

1. User Registration: Allow users to create an account and provide necessary information like name, contact details, and delivery address.

2. Snack Selection: Provide a user-friendly interface to browse and select from a wide variety of snacks, including options for different dietary preferences or restrictions.

3. Customization: Allow users to customize their snack orders, such as choosing toppings, sauces, or portion sizes, to cater to individual preferences.

4.2 Non-functional requirements :

1. Performance: The app should be fast and responsive, allowing users to navigate through snack options, customize their orders, and complete the payment process smoothly without any significant delays.

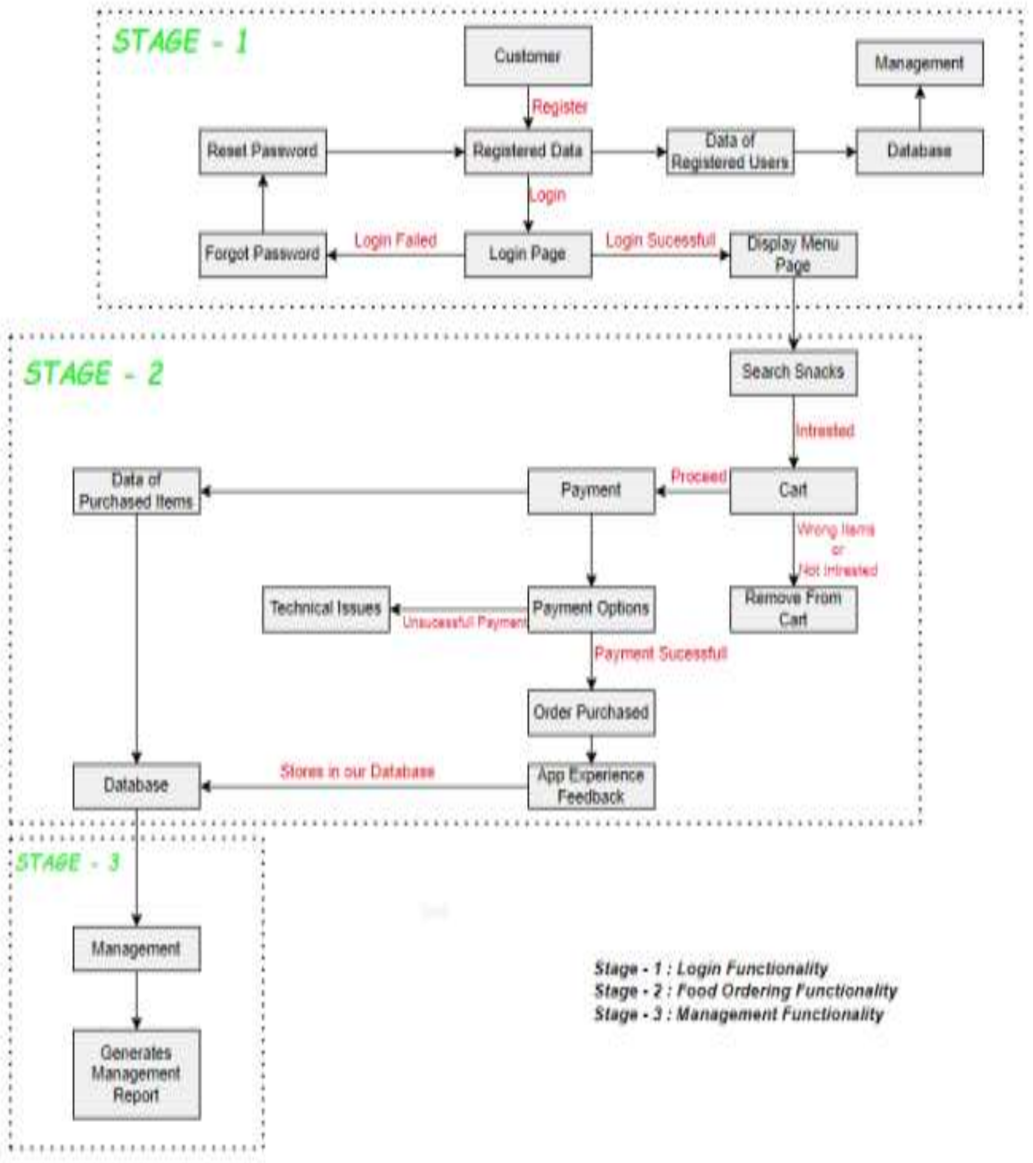
2. Security: The app should prioritize the security of user information, including personal details, payment data, and order history. It should implement robust security measures to protect against unauthorized access or data breaches.

3. Usability: The app should have an intuitive and user-friendly interface, making it easy for users to browse snacks, customize orders, and complete the payment process without confusion or frustration.

4. Scalability: The app should be designed to handle a growing number of users and orders without compromising performance or user experience. It should be able to scale up as the user base expands.

5. Data Flow Diagram & User Stories

Data Flow Diagrams:



User Stories
Use the below template to list all the user stories for the product

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user, I want to create a Snack Squad account with phone number / Email so that I can order snacks.	I can access my account / dashboard	High	Sprint-1
		USN-2	As a user, I will receive confirmation email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
		USN-3	As a user, I want to save password and username.	I can register & access the dashboard with Facebook Login	High	Sprint-2
		USN-4	As a user, I want the app to automatically verify OTPs received on my phone.	I can auto fill the OTPs received	Medium	Sprint-1
	Login	USN-5	As a user, I can log into the application by entering email & password	I can easily login to application	High	Sprint-1
	Dashboard	USN-6	As a user with an account, I'd like to have a custom dashboard to see and handle all my account info and keep track of important measurements.	I can handle my account info.		
Customer (Web user)		USN-7	As a web customer, I want to securely access my account on the web interface.	I can access my account on web interface.	High	Sprint-2
Customer (Web user)		USN-8	As a web customer, I'd like to be able to check out all the nitty-gritty details about a particular snack on the website, such as its price and ingredients.	I can check all details about snacks.	High	Sprint-1
Customer Care Executive		USN-9	As a customer care representative, I want to have access to a user-friendly dashboard or interface for managing customer inquiries and issues.	I can access user-friendly dashboard to user.	High	Sprint-1
Administrator		USN-10	As an administrator, my objective is to oversee and address any customer grievances or concerns pertaining to their orders in a timely and efficient manner.	I can address any customer grievances to their order in efficient manner	High	Sprint-1
Administrator		USN-11	As an administrator, I aim to create comprehensive reports on sales data, top-selling snacks, and customer feedback.	I can create comprehensive reports on sales data	High	Sprint-2

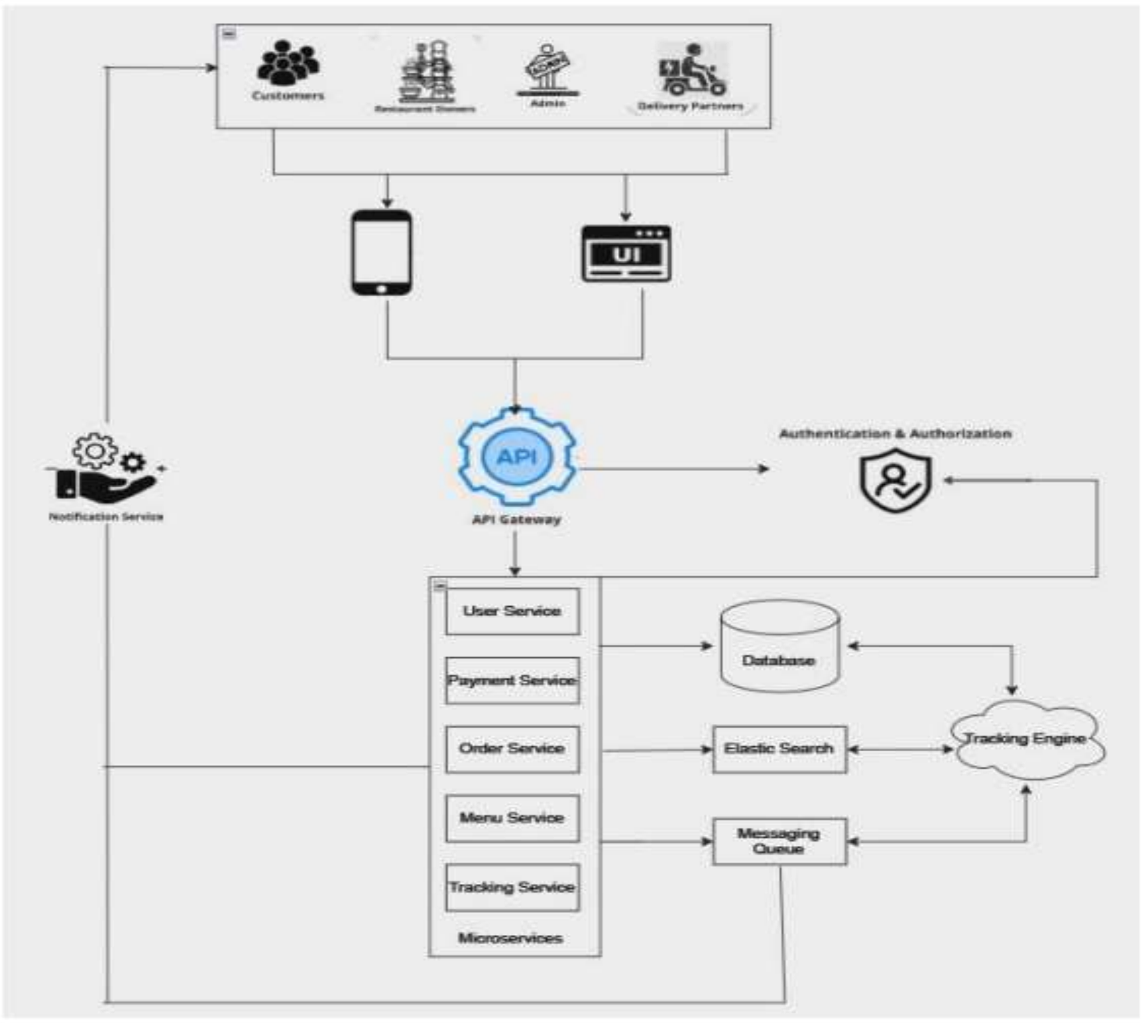
5.2.Solution Architecture :

Solution Architecture:

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions. Its goals are to:

- **Security and Privacy**
- **Upkeep and assistance**
- **Database and Data Storage**
  - ➢ Customer Data
  - ➢ Menu and Inventory Data
- **Client-Side Components**
  - ➢ Mobile App
  - ➢ Web App
- **Server-Side Components**
  - ➢ Authentication and Authorization
  - ➢ Notification Service
- **Scalability**

Solution Architecture Diagram:



Proposed Solution:

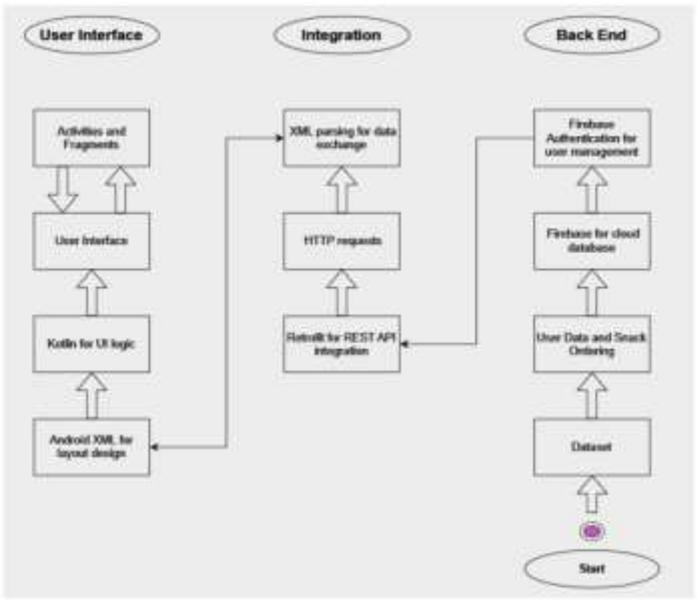
Proposed Solution Template:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	It lacks efficiency, making it difficult for customers to quickly place their orders. Additionally, the limited snack options result in a lack of variety, restricting customers' choices based on their preferences and dietary requirements. It's also challenging for customers with specific dietary needs to make informed choices since the platform doesn't provide sufficient dietary information about the snacks. These are definitely areas that could use improvement to enhance the overall snack ordering experience!
2.	Idea / Solution description	We'll make the snack ordering app awesome by adding user profiles, a recommendation system, real-time tracking, multiple payment options, an in-app wallet, language support, a review system, special promotions, delivery optimization, eco-friendly packaging, healthier snack options, group orders, offline mode, allergen alerts, and augmented reality. We'll also collaborate with local partners for a wide range of options. The ordering process will be straightforward, and you can easily browse snacks with detailed dietary information. We'll even have different offers and discounts. It's going to be amazing!

3.	Novelty / Uniqueness	Users can create their own snacks virtually, choosing ingredients, flavors, and toppings, giving a unique and personalized experience. Offering multiple language options further enhances accessibility and inclusivity.
4.	Social Impact / Customer Satisfaction	The growth of snack delivery services can open up employment opportunities for delivery drivers, customer service personel, and kitchen staff. Effective order management and inventory tracking in these apps can minimize food wastage by ensuring snacks are prepared and delivered according to real-time demand. Customers have the freedom to browse menus, check out reviews, and see ratings to make well-informed choices. And when it comes to timely delivery, it truly enhances the overall experience.
5.	Business Model (Revenue Model)	The app earns money by taking a commission from restaurants for each order, which is then divided among employees. Additionally, users are charged a delivery fee for doorstep convenience, and that fee is given to the delivery person.
6.	Scalability of the Solution	To optimize our app's performance,we are considering a cloud-based architecture like AWS, Azure, or Google Cloud. This allows for scalability, load balancing, and flexible resource allocation. Implement load balancers to distribute network traffic across multiple backend servers, preventing bottlenecks. For effective data handling, we use a scalable database system with more features .

Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2


User Interface
Integration
Back End
Activities and Fragments
XML parsing for data exchange
Firebase Authentication for user management
User Interface
HTTP requests
Firebase for cloud database
Kotlin for UI logic
Retrofit for REST API integration
User Data and Snack Ordering
Android XML for layout design
Dataset
Start

Guidelines:

1. Include all the processes (As an application logic / Technology Block)
2. Provide infrastructural demarcation (Local / Cloud)
3. Indicate external interfaces (third party API's etc.)
4. Indicate Data Storage components / services

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	The user interface provides a visually appealing and intuitive design that makes it easy for users to browse menus.	XML Layouts, Material Design guidelines
2.	Mobile App	The mobile app allows users to browse menus, customize orders and it also offers convenience, a user-friendly interface.	Kotlin, Android Studio
3.	Database	A database will help app to store and manage various types of information, such as user profiles, menu items, orders, payment details, and delivery addresses.	Room Database, SQLite
4.	Navigation	Navigation helps users move smoothly between different sections, such as browsing menus, selecting items, customizing orders, and checking out.	Navigation Component, Deep Linking
5.	Authentication	Authentication is used to ensure secure access to user accounts and it typically involves the use of login credentials, such as a username and password, to verify the identity of the user.	Firebase Authentication, OAuth2
6.	Push Notifications	The push notifications is used to send timely updates and notifications to users this also include order confirmations, order status updates, delivery tracking, promotional offers, and special offers .	Firebase Cloud Messaging, Pusher, or OneSignal

Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	User Profiles	user profiles are used to personalize the experience for each user which include storing and utilizing information such as user preferences, dietary restrictions, favorite restaurants, previous orders, and delivery addresses.	Firebase Realtime Database, SQLite
2.	Cart Management	cart management is used to allow users to add, remove, and modify items in their cart before placing an order.	Local data storage, LiveData
3.	Checkout and Payment	checkout and payment characteristics are used to facilitate the seamless completion of an order. Users can review their order details, select a payment method.	Payment gateways (e.g., Stripe, PayPal), In-App Billing
4.	Location-Based Services	Location-based are used to provide users with accurate restaurant recommendations based on their current location.	Android Location Services, Google Maps API
5.	Feedback Mechanism	Feedback mechanism is used to gather user reviews and ratings for restaurants and delivery services. This helps other users make informed decisions .	Firebase Firestore, Analytics

Project planning Phase:

Product Backlog, Sprint Schedule, and Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I want to create a Snack Squad account with phone number / Email so that I can order snacks.	9	High	Nishanth Reddy
Sprint-1		USN-2	As a user, I will receive confirmation email once I have registered for the application	7	High	Sainadh
Sprint-2		USN-3	As a user, I want to save password and username.	8	High	Nishanth Reddy
Sprint-1		USN-4	As a user, I want the app to automatically verify OTPs received on my phone.	4	Medium	Uday
Sprint-1	Login	USN-5	As a user, I can log into the application by entering email & password	6	High	Sainadh
Sprint-1	Dashboard	USN-6	As a user with an account, I'd like to have a custom dashboard to see and handle all my account info and keep track of important measurements.	8	High	Lasya
Sprint-2		USN-7	As a web customer, I want to securely access my account on the web interface.	6	High	Uday

Sprint-1		USN-8	As a web customer, I'd like to be able to check out all the nitty-gritty details about a particular snack on the website, such as its price and ingredients.	8	High	Uday
Sprint-1		USN-9	As a customer care representative, I want to have access to a user-friendly dashboard or interface for managing customer inquiries and issues.	7	High	Lasya
Sprint-1		USN-10	As an administrator, my objective is to oversee and address any customer grievances or concerns pertaining to their orders in a timely and efficient manner.	8	High	Sainadh
Sprint-2		USN-11	As an administrator, I aim to create comprehensive reports on sales data, topselling snacks, and customer feedback.	8	High	Lasya

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	57	8 Days	28 Oct 2023	4 Nov 2023	57	4 Nov 2023
Sprint-2	22	4 Days	05 Nov 2023	08 Nov 2023	22	08 Nov 2023

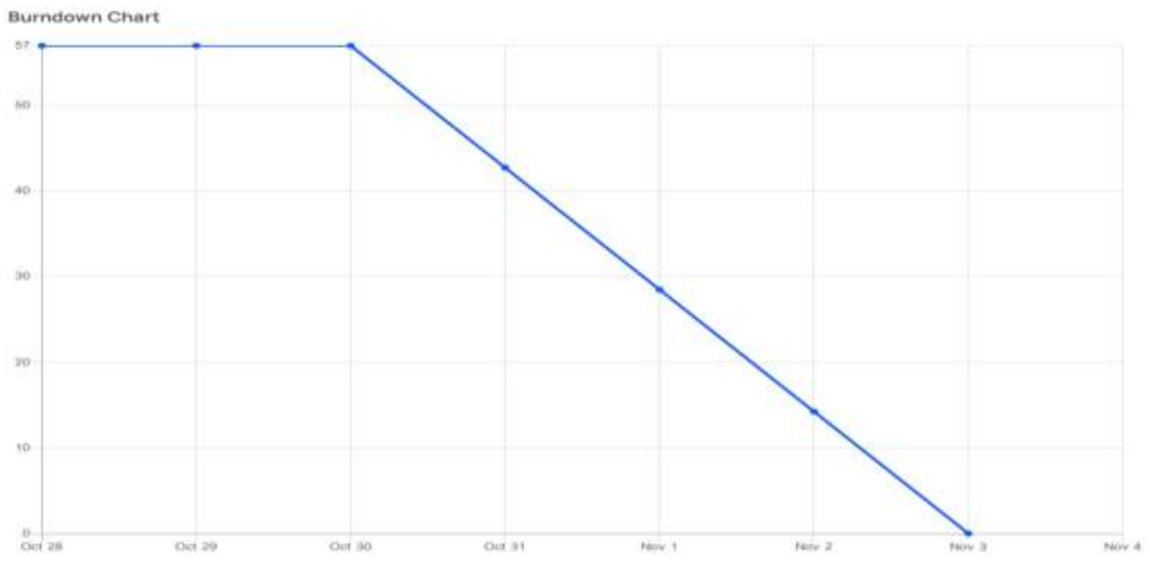
Velocity:

Imagine we have a 10-days sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

AV= 57/8 = 7.12

Burndown Chart:

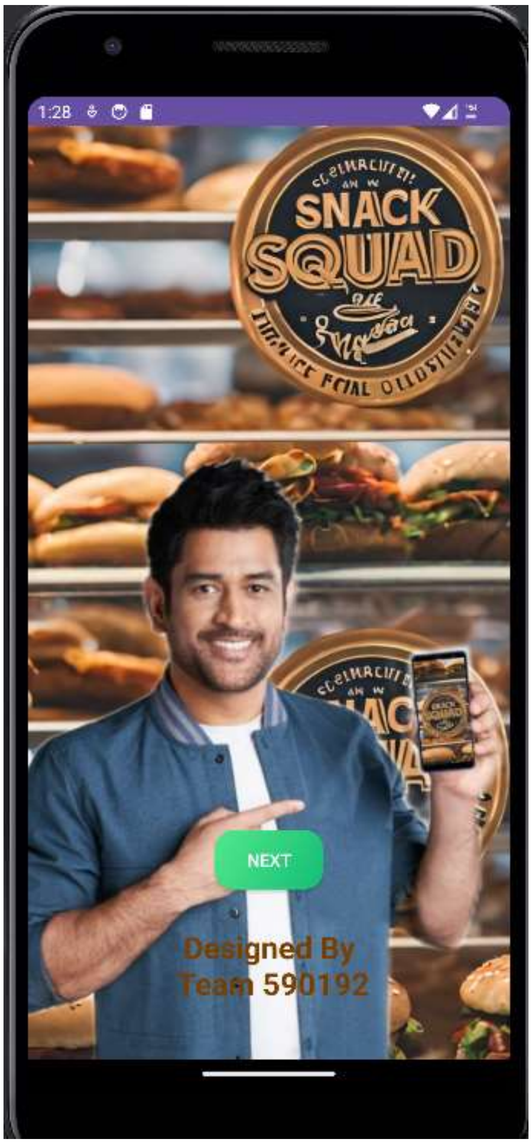
A burndown chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

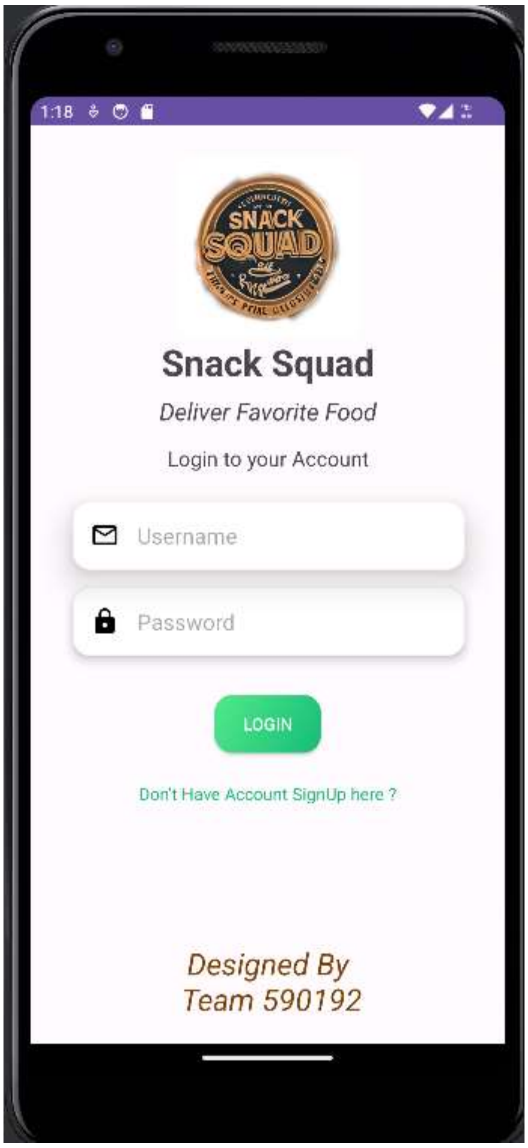


7.Coding and Solution Phase

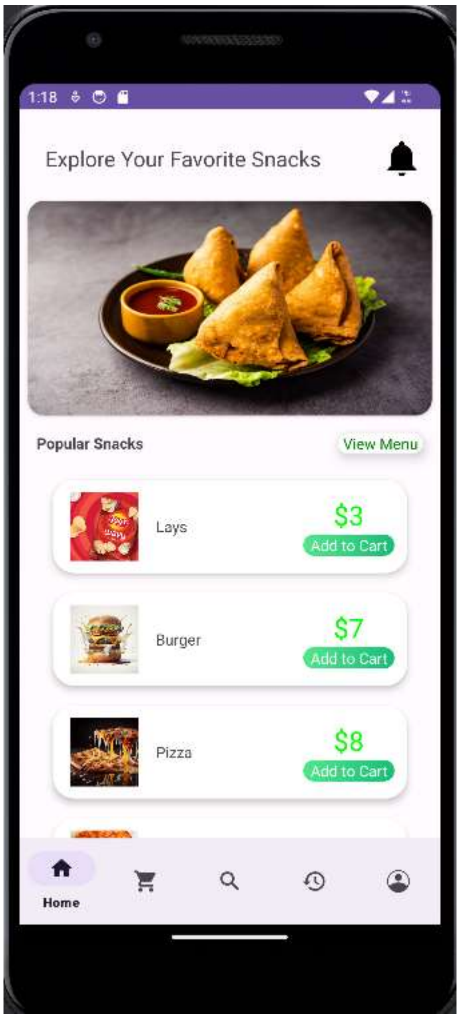
7.1 Feature 1

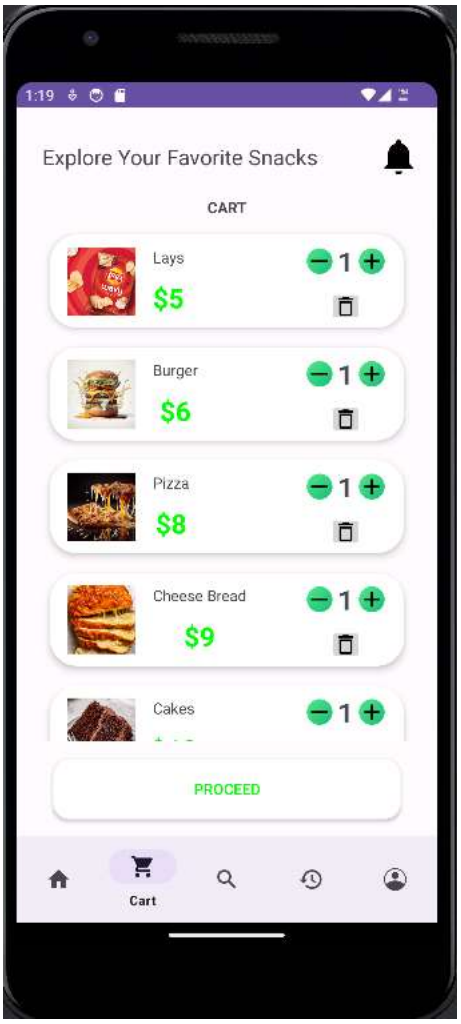
Users can login into the app after registering by clicking on signup option.



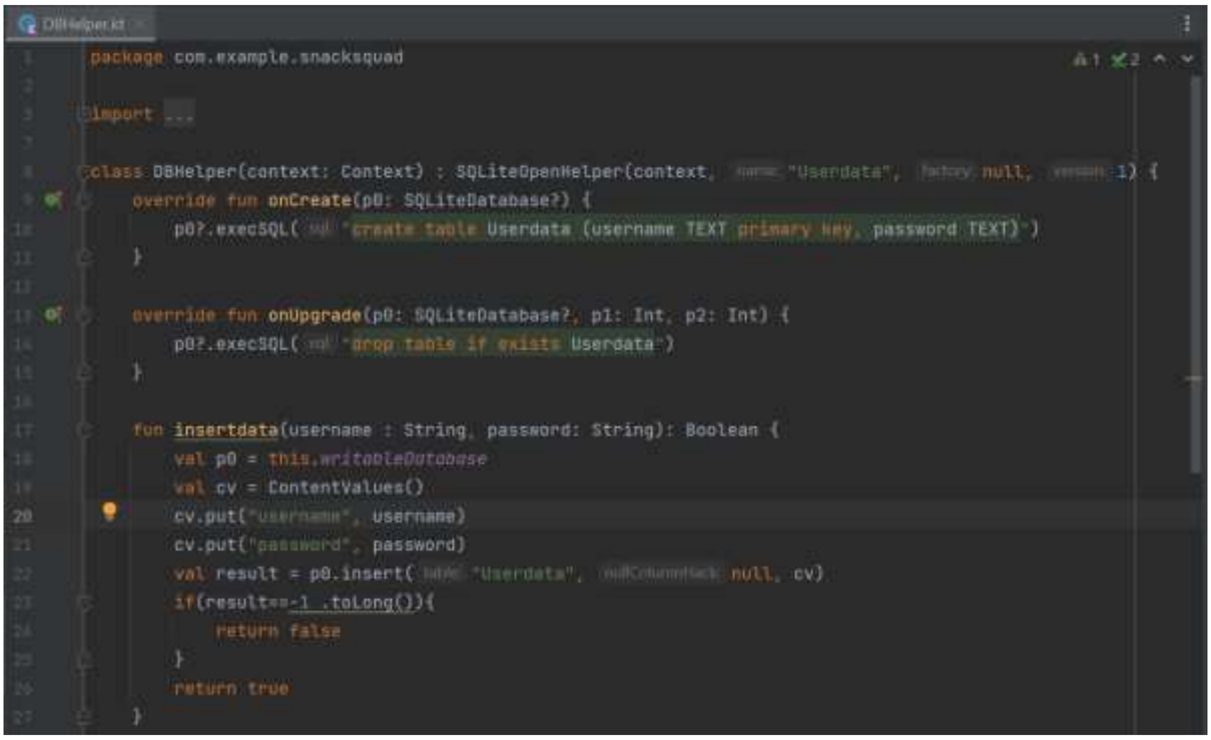


7.2 Feature 2





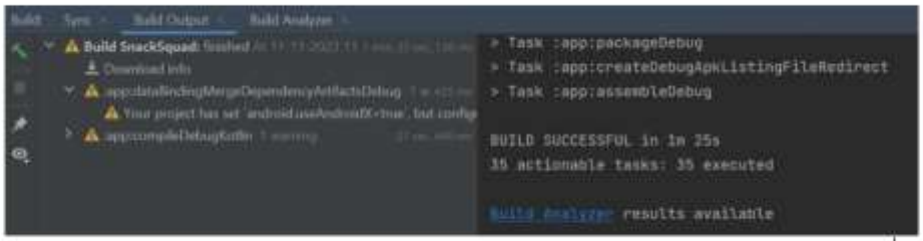
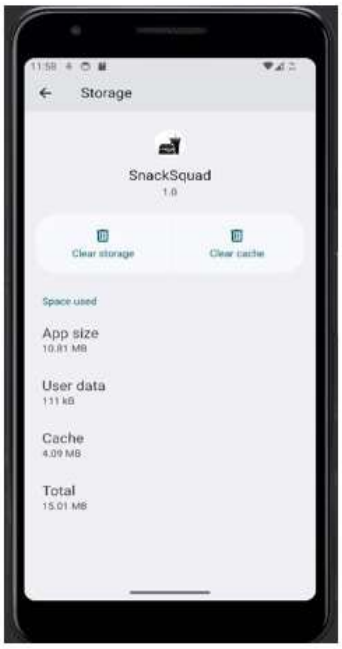
7.3 Feature 3-Database Schema

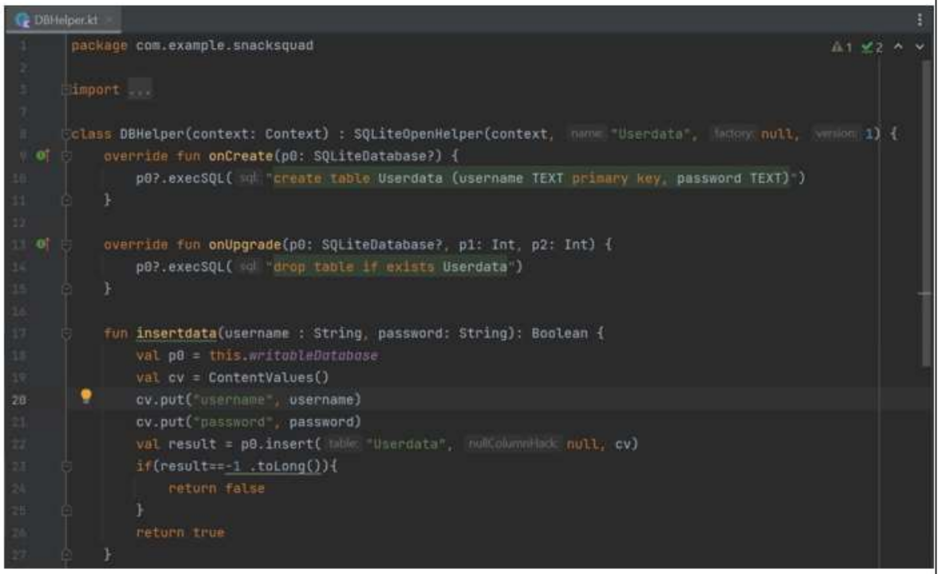


DBHelper.kt

```
package com.example.snacksquad

import ...

class DBHelper(context: Context) : SQLiteOpenHelper(context, name: "Userdata", factory: null, version: 1) {
    override fun onCreate(p0: SQLiteDatabase?) {
        p0?.execSQL( sql: "create table Userdata (username TEXT primary key, password TEXT)")
    }

    override fun onUpgrade(p0: SQLiteDatabase?, p1: Int, p2: Int) {
        p0?.execSQL( sql: "drop table if exists Userdata")
    }

    fun insertdata(username : String, password: String): Boolean {
        val p0 = this.writableDatabase
        val cv = ContentValues()
        cv.put("username", username)
        cv.put("password", password)
        val result = p0.insert( table: "Userdata", nullColumnHack: null, cv)
        if(result==-1 .toLong()){
            return false
        }
        return true
    }
```

8.PERFORMANCE TESTING

S.No.	Parameter	Values	Screenshot
1.	Metrics	App Launch Time- Screen Render Time- Code Quality-	 Build Sync Build Output Build Analyzer Build SnackSquad: finished Download info app:dataBindingMergeDependencyArtifactsDebug Your project has set 'android.useAndroidX=true', but config app:compileDebugKotlin 1 warning > Task :app:packageDebug > Task :app:createDebugApkListingFileRedirect > Task :app:assembleDebug BUILD SUCCESSFUL in 1m 25s 35 actionable tasks: 35 executed Build Analyzer results available
2.	Usage	App Size- Customer Experience-	 11:58 Storage SnackSquad 1.0 Clear storage Clear cache Space used App size 10.81 MB User data 111 kB Cache 4.09 MB Total 15.01 MB

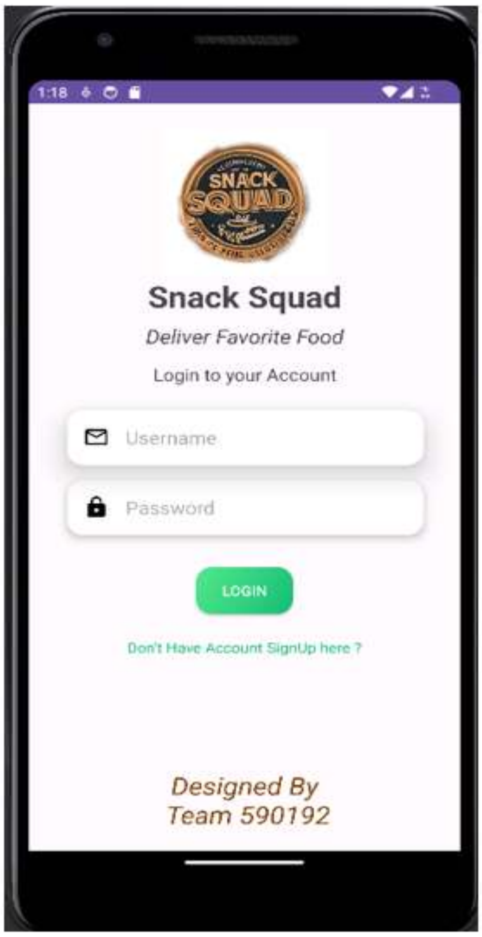
3.	Performanc e	Error and Crash Rates- Database Query Performance -	

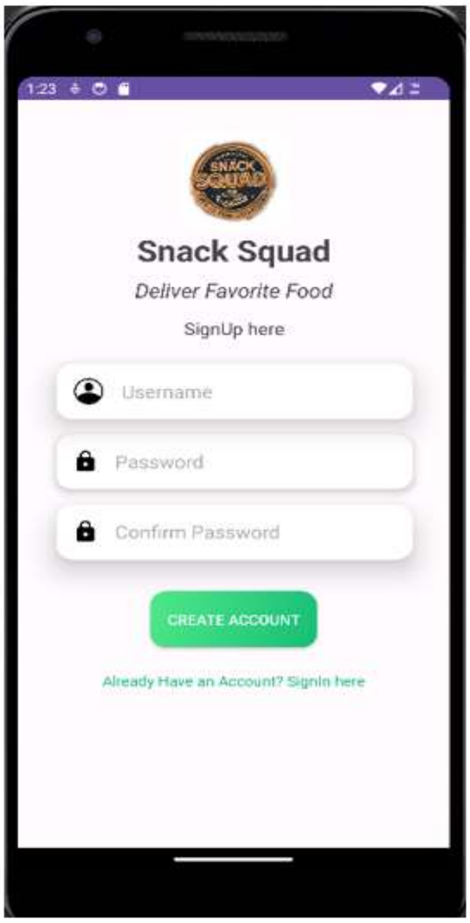
```
package com.example.snacksquad

import ...

class DBHelper(context: Context) : SQLiteOpenHelper(context, name: "Userdata", factory: null, version: 1) {
    override fun onCreate(p0: SQLiteDatabase?) {
        p0?.execSQL( sql: "create table Userdata (username TEXT primary key, password TEXT)")
    }

    override fun onUpgrade(p0: SQLiteDatabase?, p1: Int, p2: Int) {
        p0?.execSQL( sql: "drop table if exists Userdata")
    }

    fun insertdata(username : String, password: String): Boolean {
        val p0 = this.writableDatabase
        val cv = ContentValues()
        cv.put("username", username)
        cv.put("password", password)
        val result = p0.insert( table: "Userdata", nullColumnHack: null, cv)
        if(result==-1 .toLong()){
            return false
        }
        return true
    }
```

9.RESULTS

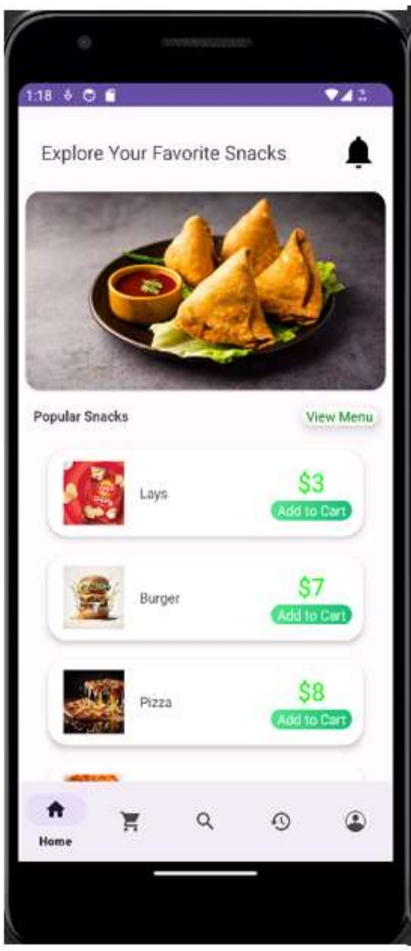
Login screen

sign up screen

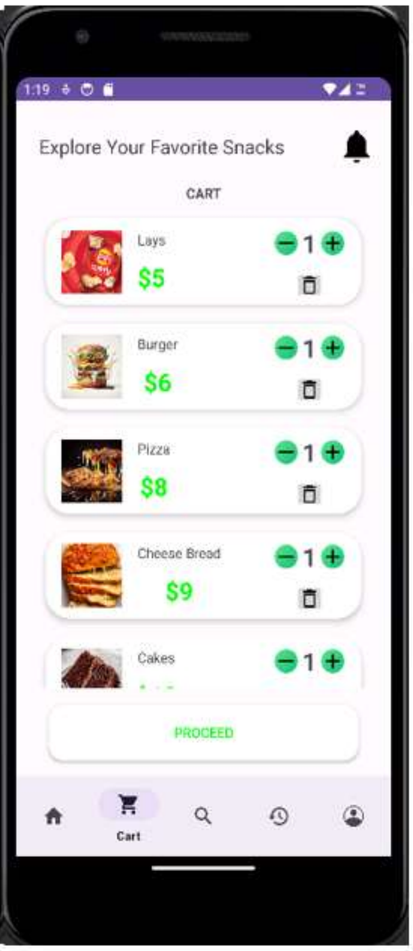




Home screen



cart screen



10.ADVANTAGES & DISADVANTAGES

ADVANTAGES

Efficiency: This app is efficient for ordering snacks! User can simply tap on phone and browse through a wide variety of snack options and can choose favorite snacks and place an order for delivery or pickup. It's a convenient and quick way to satisfy users cravings.

Clarity: By checking the menu, food items, offers, reviews, and ratings, customers can make a decision on whether to order a particular food or not. This helps them ensure a good taste experience.

Engagement: The menu, special offers, wide range of variety with images, and personalized recommendations based on their preferences and order history all contribute to keeping users engaged. Additionally, features like adding items to favorites and easy checkout options enhance the overall engagement. By this features apps can ensure that users stay hooked and come back for more delicious treats.

Convenience: Snack ordering apps are incredibly convenient to use. App offer clear navigation with helpful arrows for directions, making it easy for users to find what they're looking for. Plus, you can order snacks anytime, regardless of the date or day of the week. It's all about satisfying those cravings whenever they strike.

Variety: This apps offer an incredible variety of delicious treats to choose from. Whether user craving burgers, pizzas, tacos, or even desserts, can find a wide range of options to satisfy taste buds.

DISADVANTAGES:

Limited personal interaction- Snack ordering apps don't give user the same personal interaction as eating at a restaurant. User miss out on the face-to-face connection and the overall atmosphere.

Technical glitches -Occasionally there might be technical issues with the app, such as crashes or delays in placing orders sometimes customer can also encoder issue like slow loading.This can be solved by restarting app.

11.CONCLUSION-

Snack ordering apps are like a snack treasure trove at fingertips. With just a few taps, user can explore a world of delicious treats, from cheesy pizzas to burgers and everything in between. These apps bring convenience and variety to snacking game, making it easier than ever to satisfy those cravings.

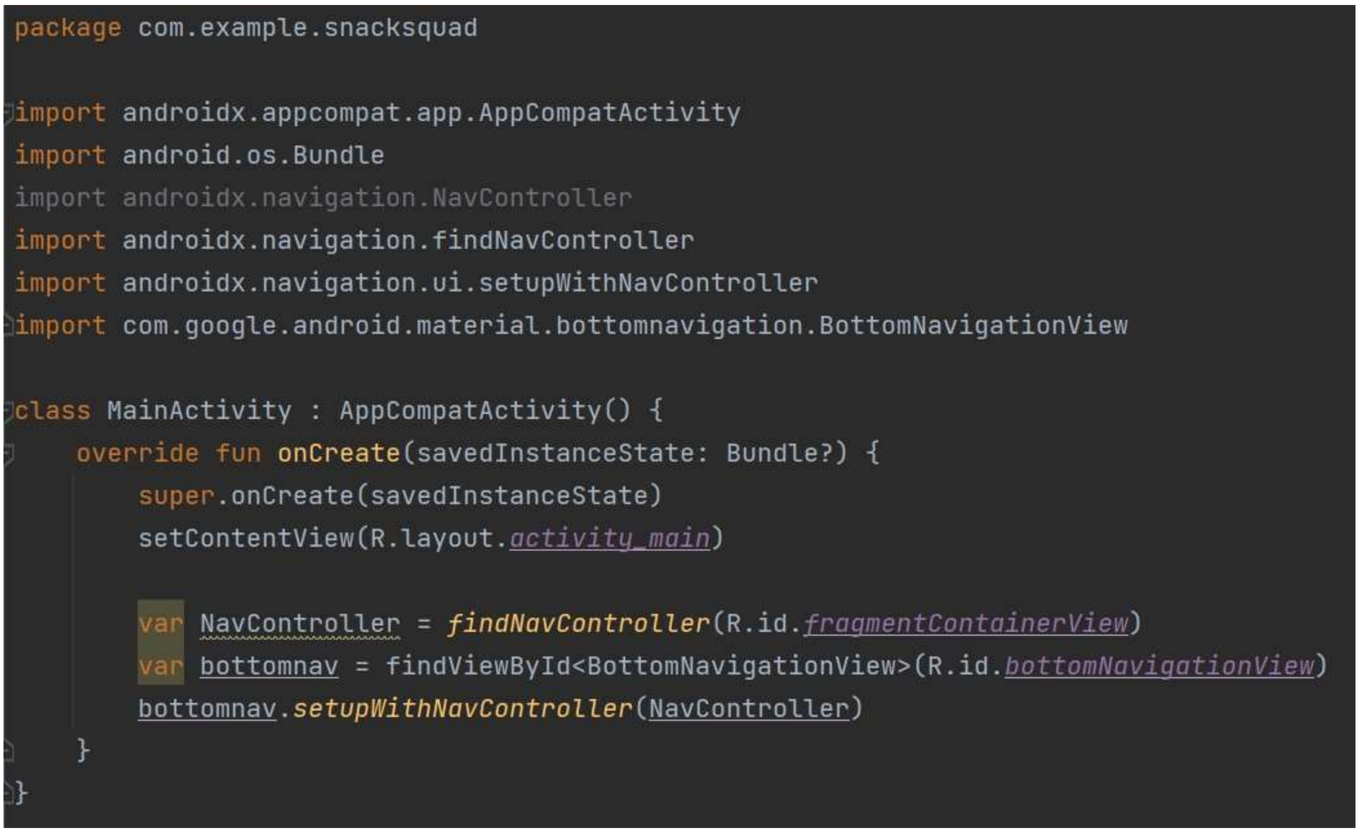
12.FUTURE SCOPE:

1. Enhanced Personalization: Snack ordering apps may utilize artificial intelligence and machine learning to provide personalized recommendations based on user preferences, dietary restrictions, and past orders.

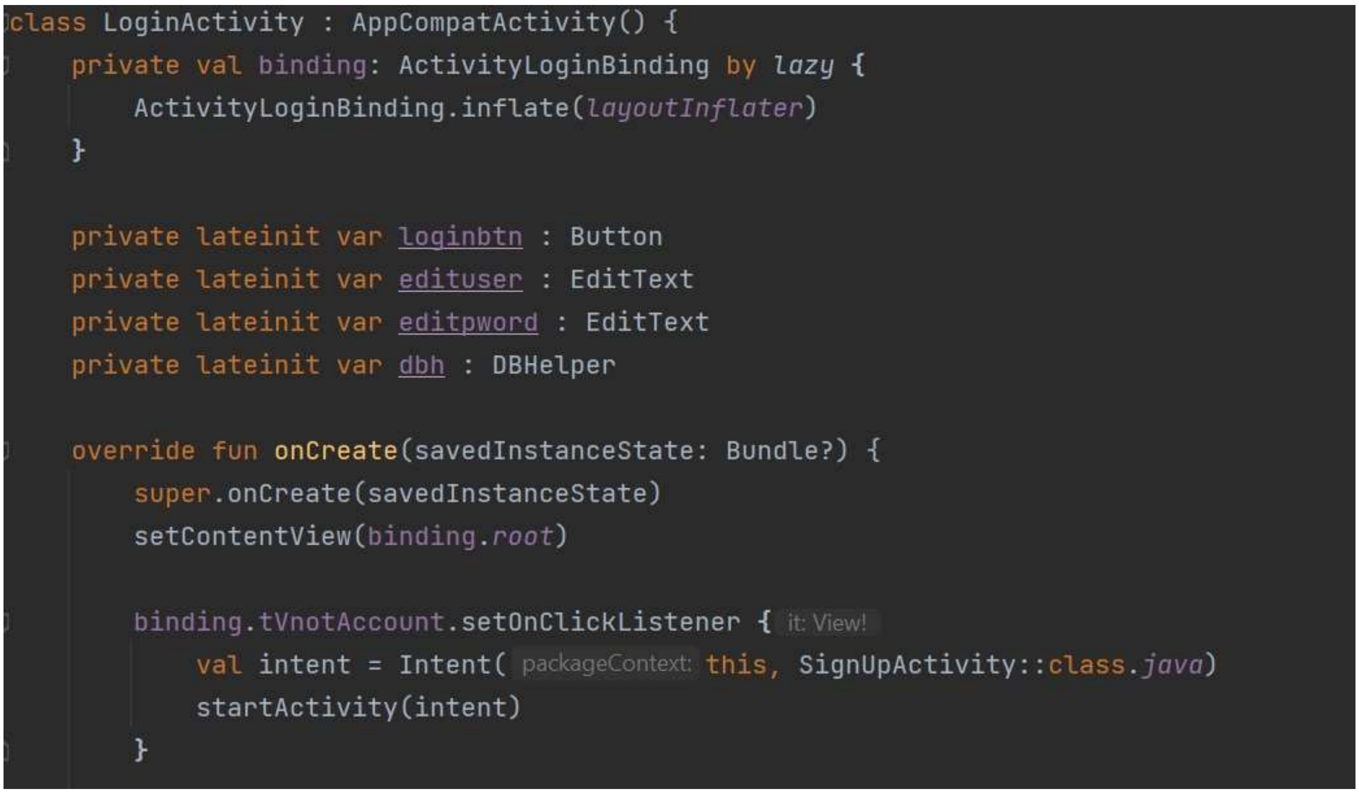
2. Integration with Smart Home Devices: Imagine being able to use voice commands to order snacks through your smart speakers or even directly from your smart refrigerator. This kind of seamless integration could make snack ordering even more convenient.

3. Augmented Reality (AR) Menu Visualization: Snack ordering apps could incorporate AR technology to allow users to visualize and interact with virtual menus. This would give customers a more immersive and engaging ordering experience.

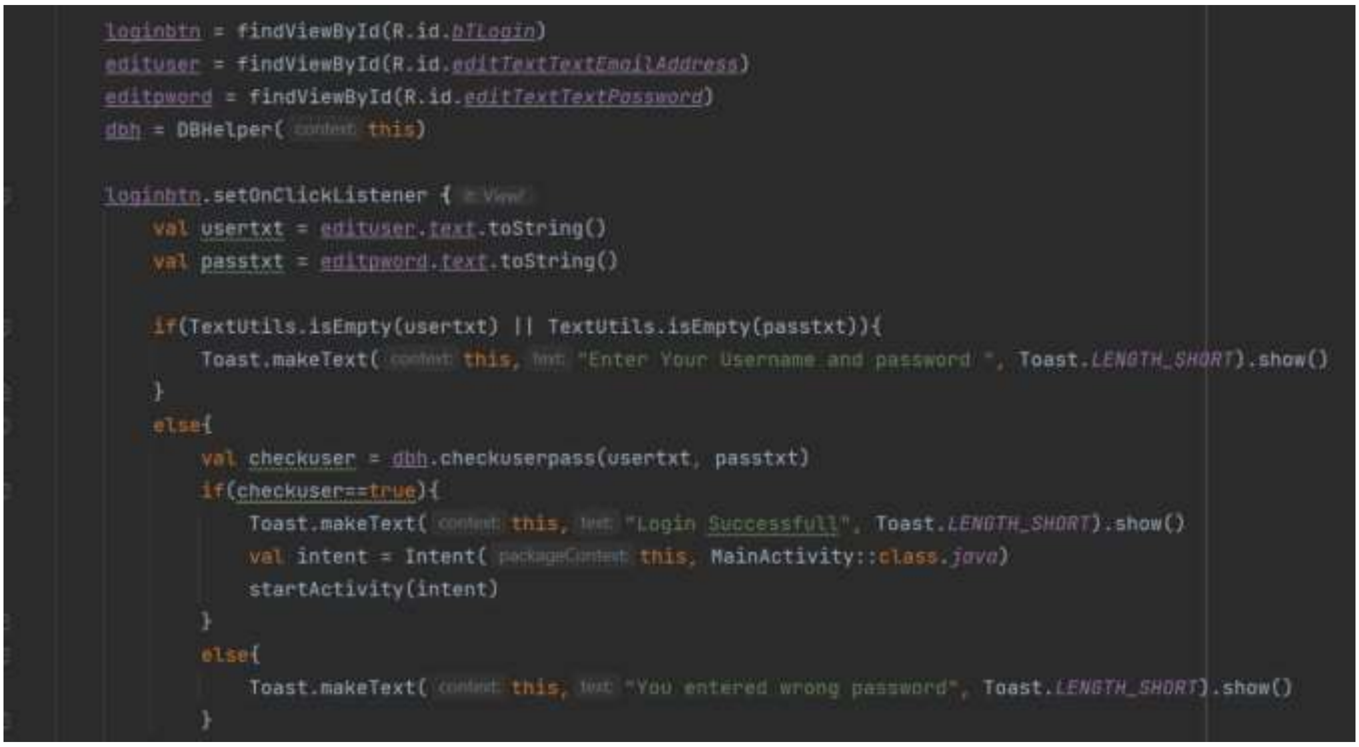
13.APPENDIX:



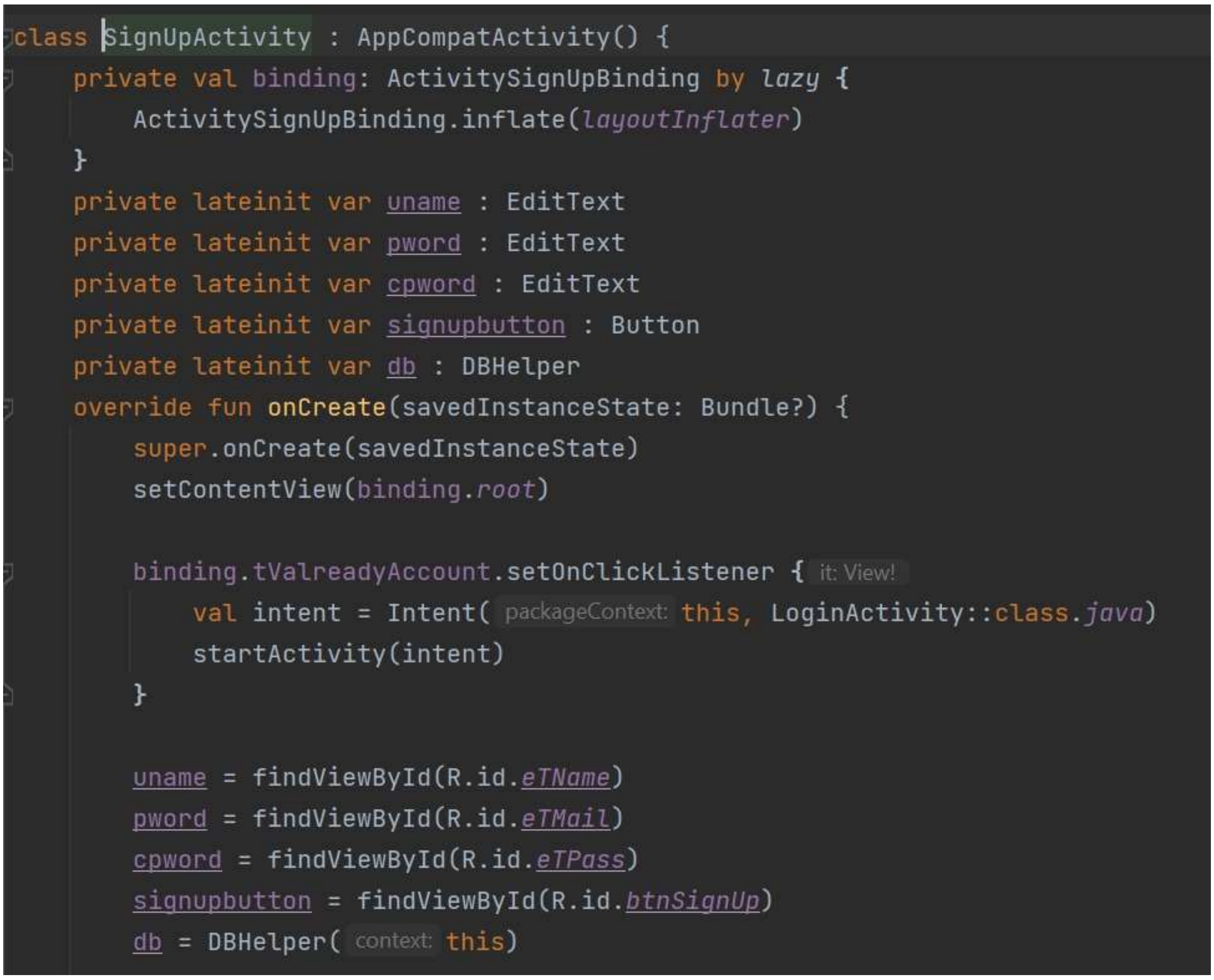
```
package com.example.snacksquad

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import androidx.navigation.NavController
import androidx.navigation.findNavController
import androidx.navigation.ui.setupWithNavController
import com.google.android.material.bottomnavigation.BottomNavigationView

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        var NavController = findNavController(R.id.fragmentContainerView)
        var bottomnav = findViewById<BottomNavigationView>(R.id.bottomNavigationView)
        bottomnav.setupWithNavController(NavController)
    }
}
```



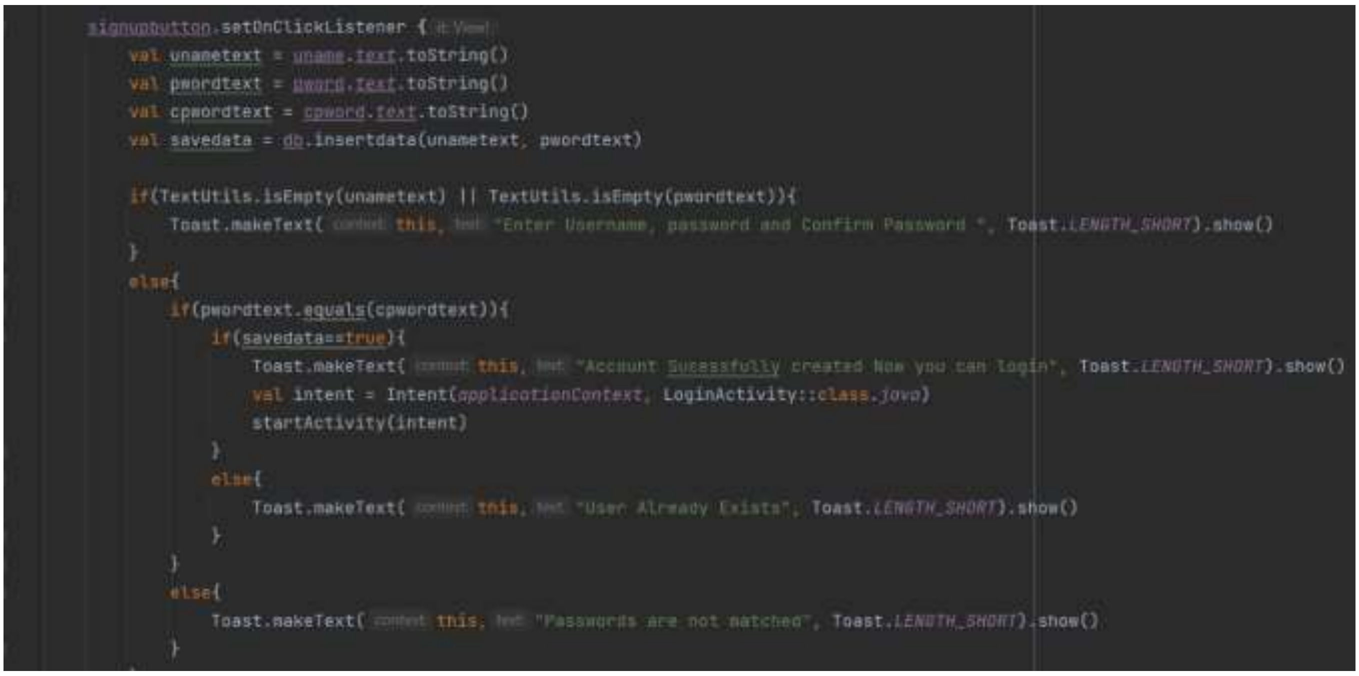
```
class LoginActivity : AppCompatActivity() {
    private val binding: ActivityLoginBinding by lazy {
        ActivityLoginBinding.inflate(layoutInflater)
    }

    private lateinit var loginbtn : Button
    private lateinit var edituser : EditText
    private lateinit var editpword : EditText
    private lateinit var dbh : DBHelper

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(binding.root)

        binding.tVnotAccount.setOnClickListener { it: View!
            val intent = Intent( packageContext: this, SignUpActivity::class.java)
            startActivity(intent)
        }
```



```
loginbtn = findViewById(R.id.bTLogin)
edituser = findViewById(R.id.editTextTextEmailAddress)
editpword = findViewById(R.id.editTextTextPassword)
dbh = DBHelper( context: this)

loginbtn.setOnClickListener { it: View!
    val usertxt = edituser.text.toString()
    val passtxt = editpword.text.toString()

    if(TextUtils.isEmpty(usertxt) || TextUtils.isEmpty(passtxt)){
        Toast.makeText( context: this, text: "Enter Your Username and password ", Toast.LENGTH_SHORT).show()
    }
    else{
        val checkuser = dbh.checkuserpass(usertxt, passtxt)
        if(checkuser==true){
            Toast.makeText( context: this, text: "Login Successfull", Toast.LENGTH_SHORT).show()
            val intent = Intent( packageContext: this, MainActivity::class.java)
            startActivity(intent)
        }
        else{
            Toast.makeText( context: this, text: "You entered wrong password", Toast.LENGTH_SHORT).show()
        }
```



```
class SignUpActivity : AppCompatActivity() {
    private val binding: ActivitySignUpBinding by lazy {
        ActivitySignUpBinding.inflate(layoutInflater)
    }
    private lateinit var uname : EditText
    private lateinit var pword : EditText
    private lateinit var cpword : EditText
    private lateinit var signupbutton : Button
    private lateinit var db : DBHelper
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(binding.root)

        binding.tValreadyAccount.setOnClickListener { it: View!
            val intent = Intent( packageContext: this, LoginActivity::class.java)
            startActivity(intent)
        }

        uname = findViewById(R.id.eTName)
        pword = findViewById(R.id.eTMail)
        cpword = findViewById(R.id.eTPass)
        signupbutton = findViewById(R.id.btnSignUp)
        db = DBHelper( context: this)
```



```
signupbutton.setOnClickListener { it: View!
    val unametext = uname.text.toString()
    val pwordtext = pword.text.toString()
    val cpwordtext = cpword.text.toString()
    val savedata = db.insertdata(unametext, pwordtext)

    if(TextUtils.isEmpty(unametext) || TextUtils.isEmpty(pwordtext)){
        Toast.makeText( context: this, text: "Enter Username, password and Confirm Password ", Toast.LENGTH_SHORT).show()
    }
    else{
        if(pwordtext.equals(cpwordtext)){
            if(savedata==true){
                Toast.makeText( context: this, text: "Account Sucessfully created Now you can login", Toast.LENGTH_SHORT).show()
                val intent = Intent(applicationContext, LoginActivity::class.java)
                startActivity(intent)
            }
            else{
                Toast.makeText( context: this, text: "User Already Exists", Toast.LENGTH_SHORT).show()
            }
        }
        else{
            Toast.makeText( context: this, text: "Passwords are not matched", Toast.LENGTH_SHORT).show()
        }
```

THANK YOU