

# **Snack Squad: A Customizable Snack Ordering and Delivery App**

## **Contents**

### **1 INTRODUCTION**

1.1 Overview

1.2 Purpose

### **2 PROBLEM DEFINITION AND DESIGN THINKING**

2.1 Empathy Map

2.2 Ideation & Brainstorming Map

### **3 RESULT**

### **4 ADVANTAGES & DISADVANTAGES**

### **5 APPLICATIONS**

### **6 CONCLUSION**

### **7 FUTURE SCOPE**

### **8 APPENDIX**

8.1 Source Code

# **INTRODUCTION**

## **1.1 OVERVIEW**

A snacks ordering app is a mobile application that allows users to browse and order a variety of snacks, drinks, and other food items from a menu. Users can create an account, log in, and select their desired items from the menu. They can then add the items to their cart and proceed to checkout. Payment can be made through the app using a secure payment gateway, and users can track the status of their order and estimated delivery time.

A project that demonstrates the use of Android Jetpack Compose to build a UI for a snack squad app. Snack Squad is a sample project built using the Android Compose UI toolkit. It demonstrates how to create a simple e-commerce app for snacks using the Compose libraries. The user can see a list of snacks, and by tapping on a snack, and by tapping on the "Add to Cart" button, the snack will be added to the cart. The user can also see the list of items in the cart and can proceed to checkout to make the purchase.

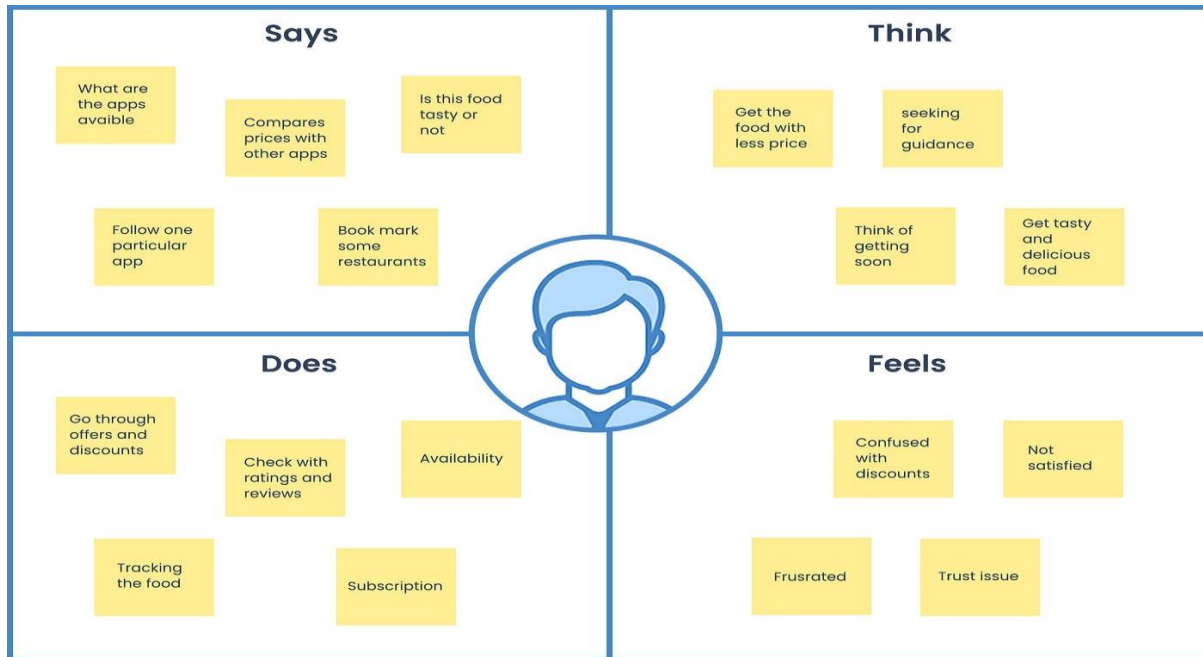
## **1.1 PURPOSE**

The purpose of a snacks ordering app is to provide users with a convenient and easy-to-use platform to order a variety of snacks, drinks, and other food items from a menu. The app eliminates the need for users to physically visit a restaurant or store to order their favorite snacks, instead, they can place their order and pay for it securely through the app.

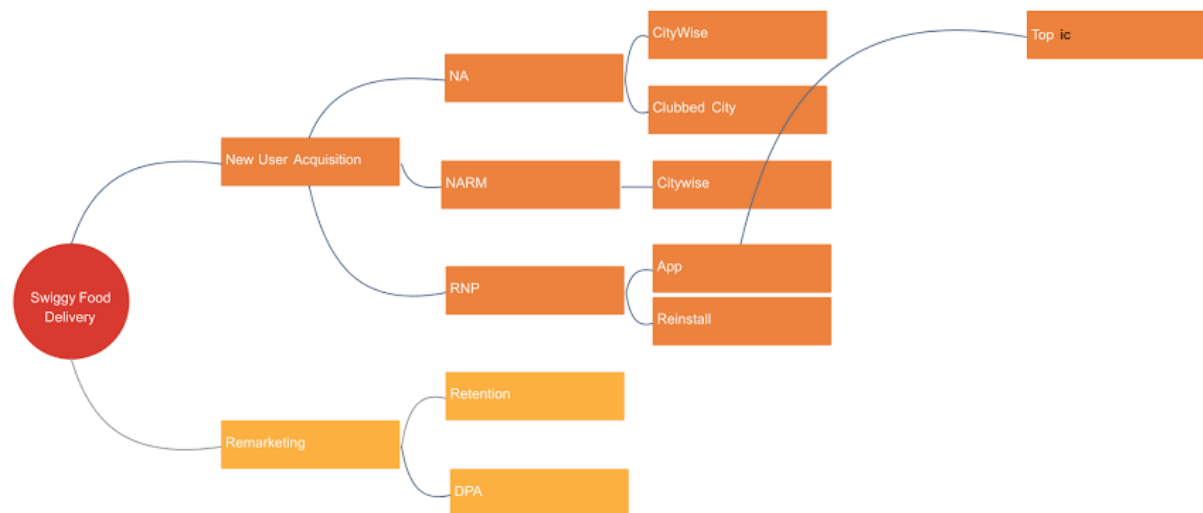
The snacks ordering app also benefits businesses by allowing them to reach a wider audience and increase their revenue by offering online ordering and delivery services. The app can help businesses streamline their operations by reducing the time and resources required to process orders and manage inventory.

## 2 Problem Definition and Design Thinking

### 2.1 EMPATHY

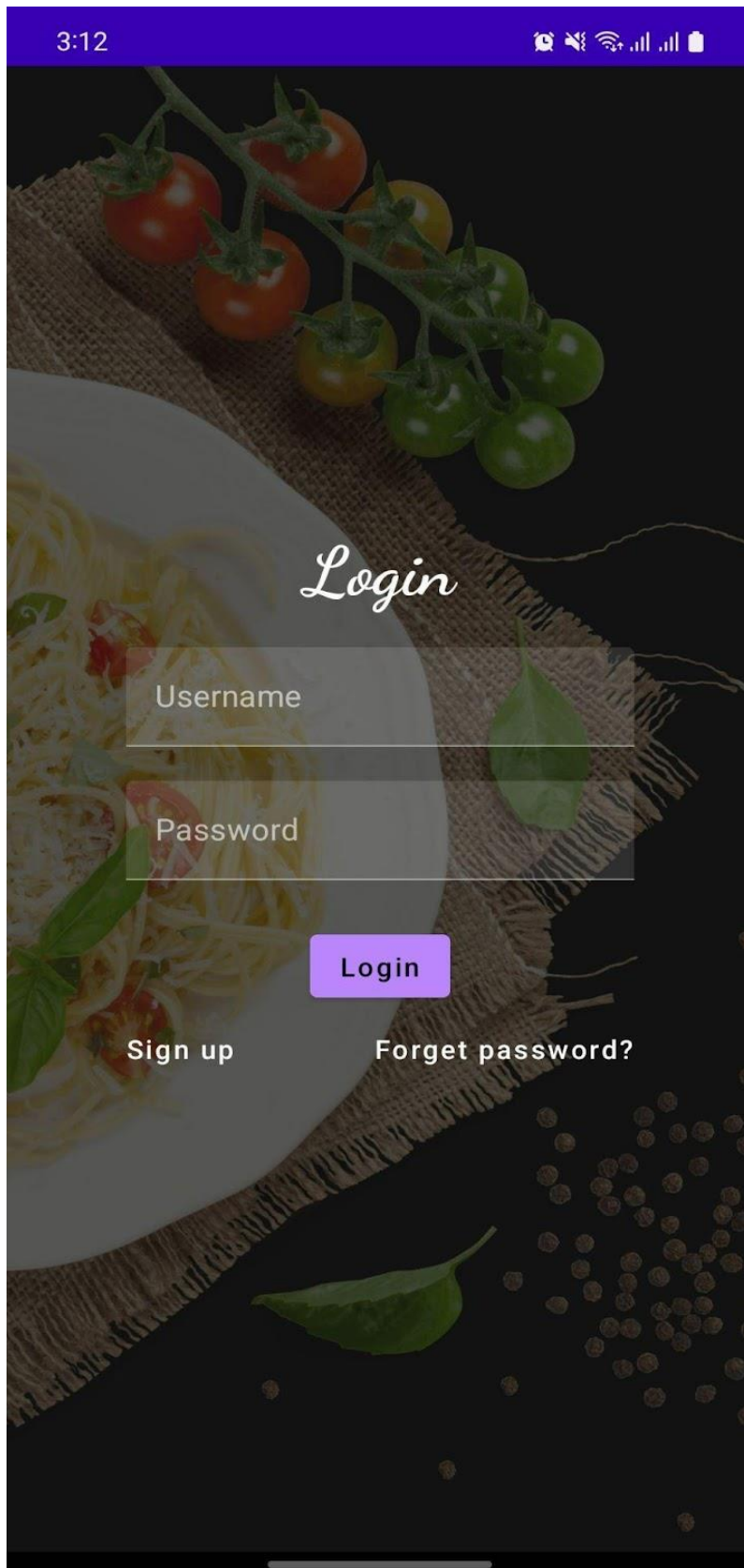


### 2.2 IDEATION AND BRAINSTROMING MAP

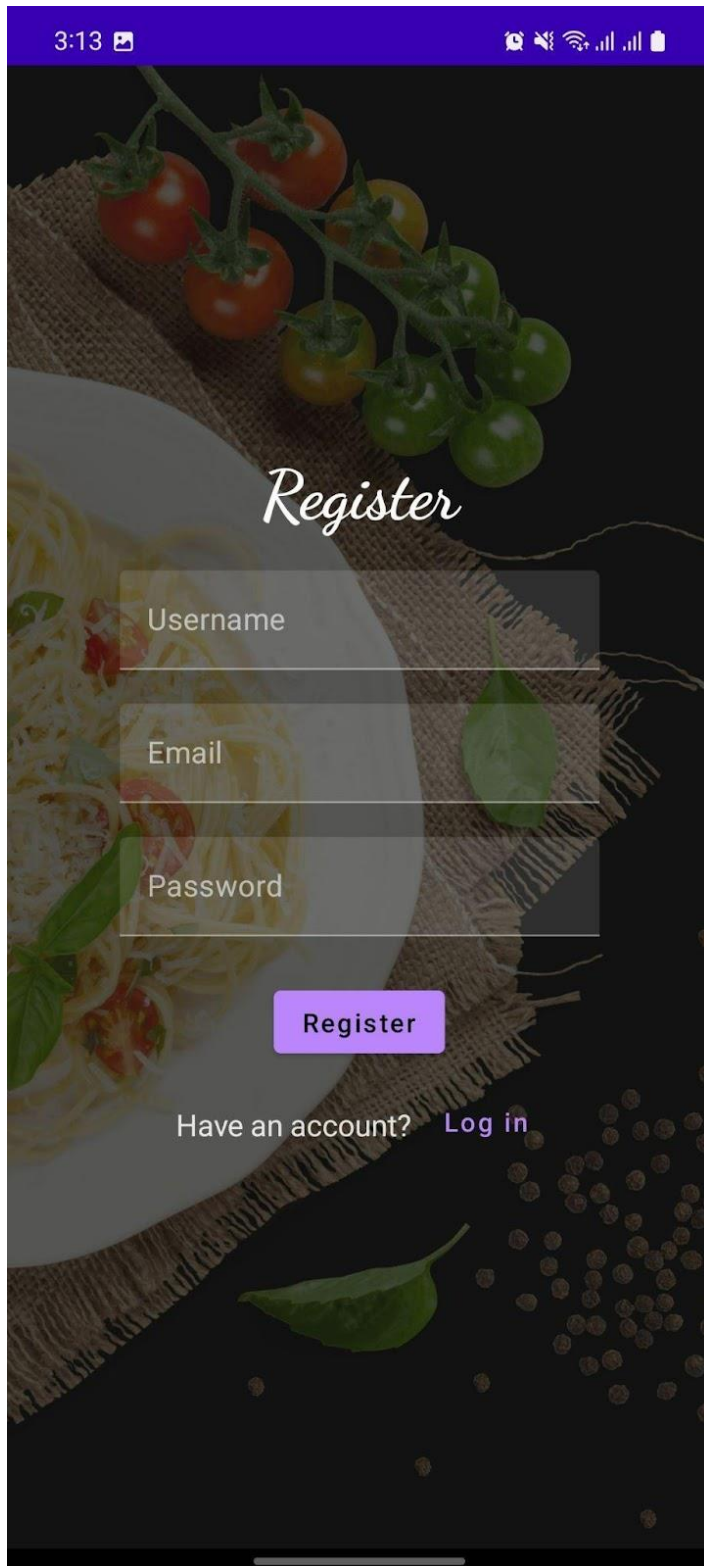


### 3 RESULT

#### SIGN IN PAGE:



## SIGN UP PAGE:

The image shows a mobile application's sign-up page. The background is a dark, high-quality photograph of a plate of spaghetti topped with tomato sauce and fresh basil leaves. A bunch of cherry tomatoes, some red and some green, is placed on a piece of burlap fabric next to the plate. The overall aesthetic is clean and appetizing. The sign-up form is centered on the screen, featuring three input fields for 'Username', 'Email', and 'Password'. Below these fields is a prominent blue 'Register' button. At the bottom of the form, there is a link that says 'Have an account? Log in', where 'Log in' is in a lighter blue color. The top of the screen shows a standard Android status bar with the time '3:13' and various system icons like signal strength, Wi-Fi, and battery level.

3:13

*Register*

Username

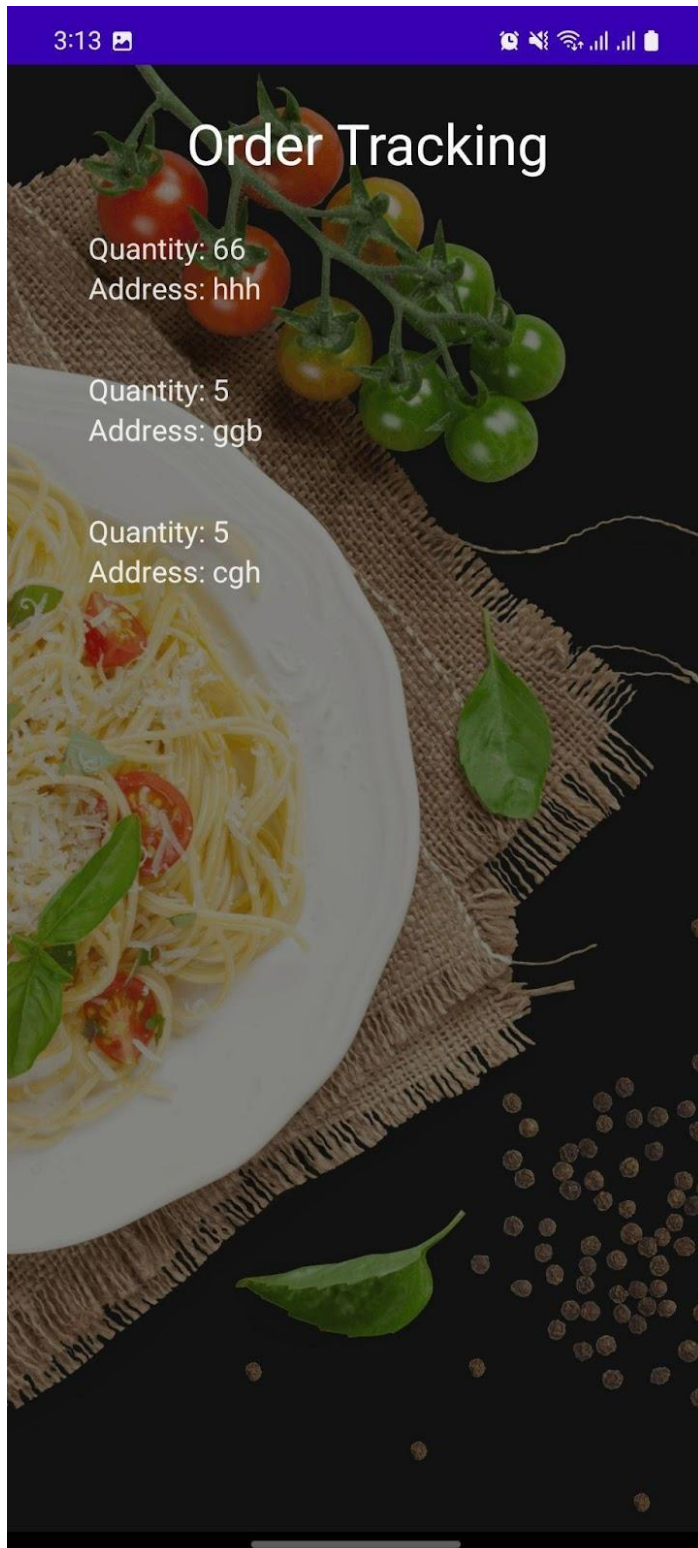
Email

Password

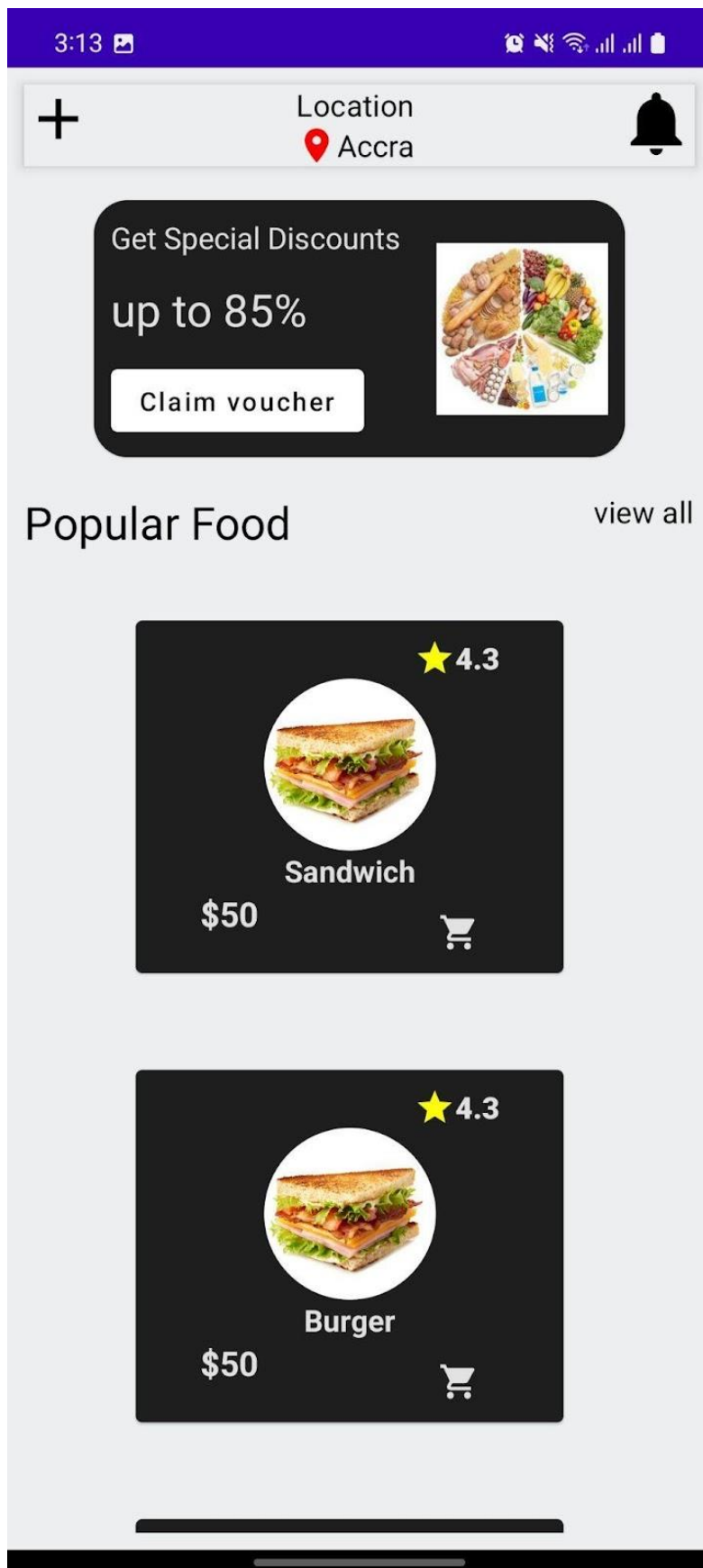
Register

Have an account? [Log in](#)

## ADMIN PAGE:

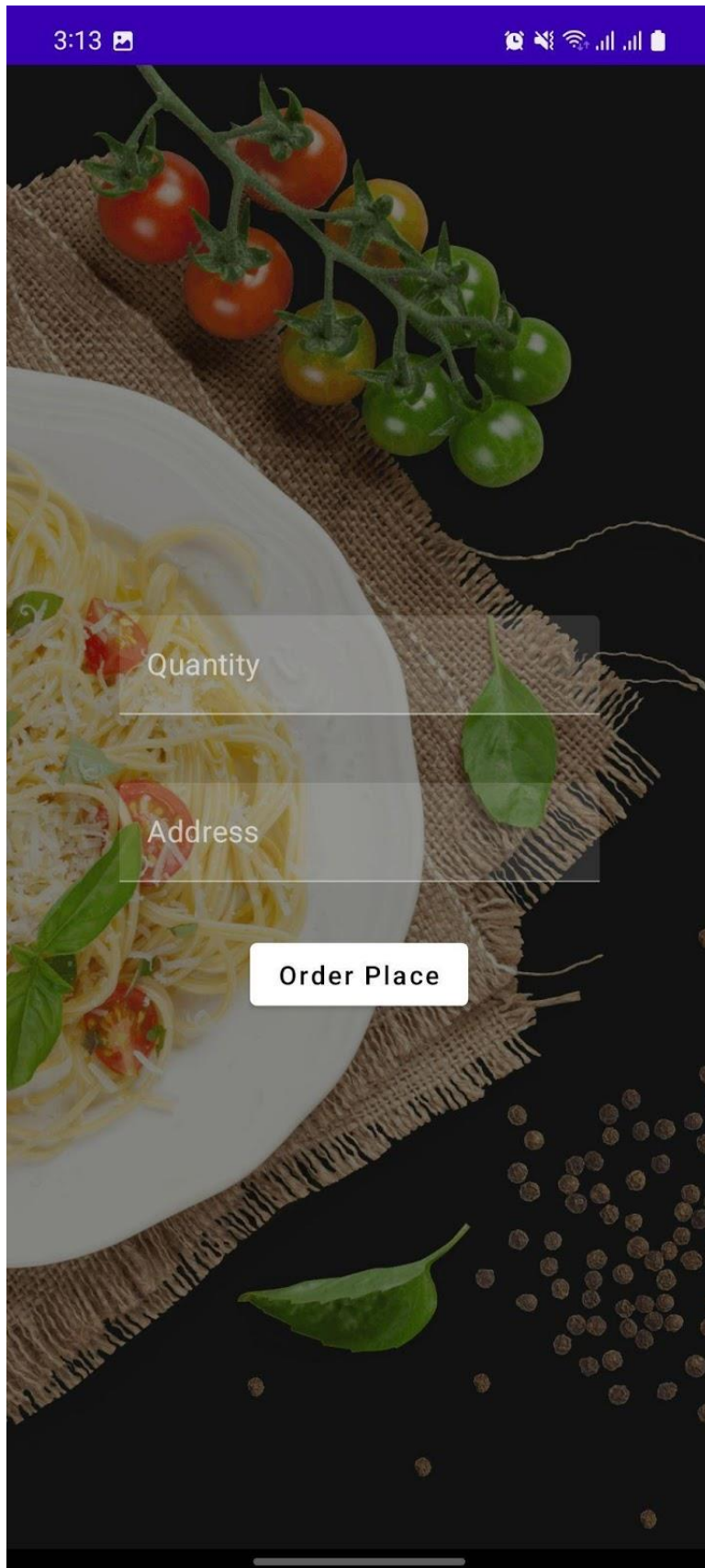


## USER PAGE:





## ORDER PAGE:



3:13

Quantity

Address

Order Place

The image shows a mobile application interface for an ordering system. The background is a high-quality photograph of a plate of spaghetti topped with tomato sauce, fresh basil leaves, and a cluster of cherry tomatoes on the vine. The plate is set on a piece of light-colored burlap fabric against a dark, textured background. Scattered brown seeds or crumbs are visible in the lower right corner. The app's status bar at the top is purple and displays the time 3:13, a camera icon, and various connectivity and battery icons. The main content area features three input fields: a 'Quantity' field, an 'Address' field, and a white 'Order Place' button. The fields are semi-transparent, allowing the background image to be seen through them.



## **4 ADVANTAGE AND DISADVANTAGE**

### **4.1 Advantages :**

1. Convenience: Users can order their favorite snacks from the comfort of their own devices, saving them time and effort.
2. Variety: A snacks ordering app can offer a wide range of snacks and food items, providing users with more choices than traditional ordering methods.
3. Efficiency: The app can streamline the ordering process, reducing wait times and improving the overall customer experience.
4. Personalization: The app can offer personalized recommendations based on the user's past orders and preferences, enhancing the user experience.
5. Easy payment: Payment can be made securely through the app, with options such as credit cards, digital wallets, or cash on delivery.

### **4.2 Disadvantages:**

1. Technical issues: Technical problems such as slow load times or app crashes can frustrate users and affect their willingness to use the app.
2. Quality control: Ensuring the quality and freshness of the snacks can be challenging when ordering online, potentially leading to dissatisfied customers.
3. Additional fees: Some apps may charge extra fees for delivery, service charges, or minimum order amounts, which can be a turn-off for some users.
4. Limited customization: Users may not be able to customize their order as much as they would be able to in person, which can be a drawback for some.
5. Dependence on internet access: Users require a reliable internet connection to use the app, which can be an issue in areas with poor connectivity.

## 5 APPLICATIONS

The application of a snacks delivery app can be vast and can be applied in a variety of ways. Here are some examples:

1. **Online Snacks Ordering:** A snacks delivery app can be used to offer online ordering services for snacks, food items, and drinks. Users can browse the menu, place their orders, and pay securely through the app.
2. **Catering Services:** A snacks delivery app can also be used to offer catering services for events such as parties, meetings, and conferences. Users can order customized snacks and food items for their guests through the app.
3. **Contactless Delivery:** A snacks delivery app can provide contactless delivery services, which have become increasingly important during the COVID-19 pandemic. Delivery personnel can drop off the order at the user's doorstep to minimize contact.
4. **Reward Programs:** A snacks delivery app can also offer reward programs for users who place frequent orders. Users can earn points or discounts on their orders, encouraging them to use the app more often.
5. **User Feedback:** A snacks delivery app can also allow users to leave feedback and ratings for the snacks and food items they order. This can help improve the quality of the snacks and enhance the overall user experience.

## **6 CONCLUSION**

With online food ordering system, restaurant an mess menu online can be set-up and customers can easily place order.

Also with a food menu online, tracking the order is done easily, it maintain customer's database and improve food delivery service.

Overall we have created website in focus of future food ordering system, this website will helpful to many people.

Implemented some modules for users feedback, we also provide post query if user not receive proper food.

## **7 FUTURE SCOPE**

The future scope of snacks delivering app is promising, as the demand for food delivery services continues to grow. Here are some potential areas of growth for snacks delivering apps:

**Personalization:** As technology continues to advance, snacks delivering apps may become more personalized, offering tailored recommendations based on the user's past orders, preferences, and dietary restrictions.

**Expansion of Services:** Snacks delivering apps may expand their services beyond food and drinks, offering additional products such as household items and groceries for delivery.

**Sustainability:** Snacks delivering apps may also focus on promoting sustainability by using eco-friendly packaging, partnering with local farmers, and reducing food waste.

## **8 APPENDIX**

### **8.1 Source Code**

## **Build Gradle:**

```
plugins {  
    id 'com.android.application'  
    id 'org.jetbrains.kotlin.android'  
}  
  
android {  
    namespace 'com.example.snackordering'  
    compileSdk 33  
  
    defaultConfig {  
        applicationId "com.example.snackordering"  
        minSdk 22  
        targetSdk 33  
        versionCode 1  
        versionName "1.0"  
  
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  
        vectorDrawables {  
            useSupportLibrary true  
        }  
    }  
  
    buildTypes {  
        release {  
            minifyEnabled false  
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'),  
'proguard-rules.pro'  
        }  
    }  
  
    compileOptions {  
        sourceCompatibility JavaVersion.VERSION_1_8  
        targetCompatibility JavaVersion.VERSION_1_8  
    }  
}
```

```

}
kotlinOptions {
    jvmTarget = '1.8'
}
buildFeatures {
    compose true
}
composeOptions {
    kotlinCompilerExtensionVersion '1.2.0'
}
packagingOptions {
    resources {
        excludes += '/META-INF/{AL2.0,LGPL2.1}'
    }
}
}

dependencies {

    implementation 'androidx.core:core-ktx:1.7.0'
    implementation 'androidx.lifecycle:lifecycle-runtime-ktx:2.3.1'
    implementation 'androidx.activity:activity-compose:1.3.1'
    implementation "androidx.compose.ui:ui:$compose_ui_version"
    implementation "androidx.compose.ui:ui-tooling-preview:$compose_ui_version"
    implementation 'androidx.compose.material:material:1.2.0'
    implementation 'androidx.room:room-common:2.5.0'
    implementation 'androidx.room:room-ktx:2.5.0'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.5'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'
    androidTestImplementation "androidx.compose.ui:ui-test-junit4:$compose_ui_version"
    debugImplementation "androidx.compose.ui:ui-tooling:$compose_ui_version"

```

```
        debugImplementation "androidx.compose.ui:ui-test-manifest:$compose_ui_version"
    }
}
```

**MANIFEST.XML:** `<?xml version="1.0" encoding="utf-8"?>`

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
```

```
<application
```

```
    android:allowBackup="true"
    android:dataExtractionRules="@xml/data_extraction_rules"
    android:fullBackupContent="@xml/backup_rules"
    android:icon="@drawable/icon"
    android:label="@string/app_name"
    android:supportsRtl="true"
    android:theme="@style/Theme.SnackOrdering"
    tools:targetApi="31">
```

```
<activity
```

```
    android:name=".AdminActivity"
    android:exported="true"
    android:label="@string/title_activity_admin"
    android:theme="@style/Theme.SnackOrdering" />
```

```
<activity
```

```
    android:name=".LoginActivity"
    android:exported="true"
    android:theme="@style/Theme.SnackOrdering">
```

```
<intent-filter>
```

```
    <action android:name="android.intent.action.MAIN" />
```

```
    <category android:name="android.intent.category.LAUNCHER" />
```

```
</intent-filter>
```

```
</activity>
```

```
<activity
```

```
    android:name=".TargetActivity"
```



```
        android:exported="false"
        android:label="@string/title_activity_target"
        android:theme="@style/Theme.SnackOrdering" />
    <activity
        android:name=".MainPage"
        android:exported="false"
        android:label="@string/title_activity_main_page"
        android:theme="@style/Theme.SnackOrdering" />
    <activity
        android:name=".MainActivity"
        android:exported="false"
        android:label="MainActivity"
        android:theme="@style/Theme.SnackOrdering" />
</application>

</manifest>
```