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

**DESCRIPTION:**

Snack Squad is a customizable snack ordering and delivery app that enables users to order snacks from a variety of local stores and have them delivered to their doorstep. The app was designed to make it easy and convenient for people to satisfy their snack cravings without having to leave their home or office.

The app works by allowing users to select their location and browse a selection of snacks from participating stores in the area. Users can customize their orders, adding or removing items, and selecting delivery or pickup options. Once the order is placed, it is sent to the store, and the Snack Squad delivery team is dispatched to fulfil the order.

One of the key features of Snack Squad is its customization options. Users can select from a variety of snack categories, including healthy options, sweet treats, and savoury snacks, and filter results based on dietary restrictions or preferences, such as vegan or gluten-free options. Additionally, users can save their favourite orders for quick and easy reordering in the future.

**PURPOSE:**

The purpose of Snack Squad is to provide a convenient and customizable way for users to order snacks from local stores and have them delivered to their doorstep. The app aims to solve the problem of snack cravings by offering a quick and easy way to order snacks without having to leave the comfort of one's home or office.

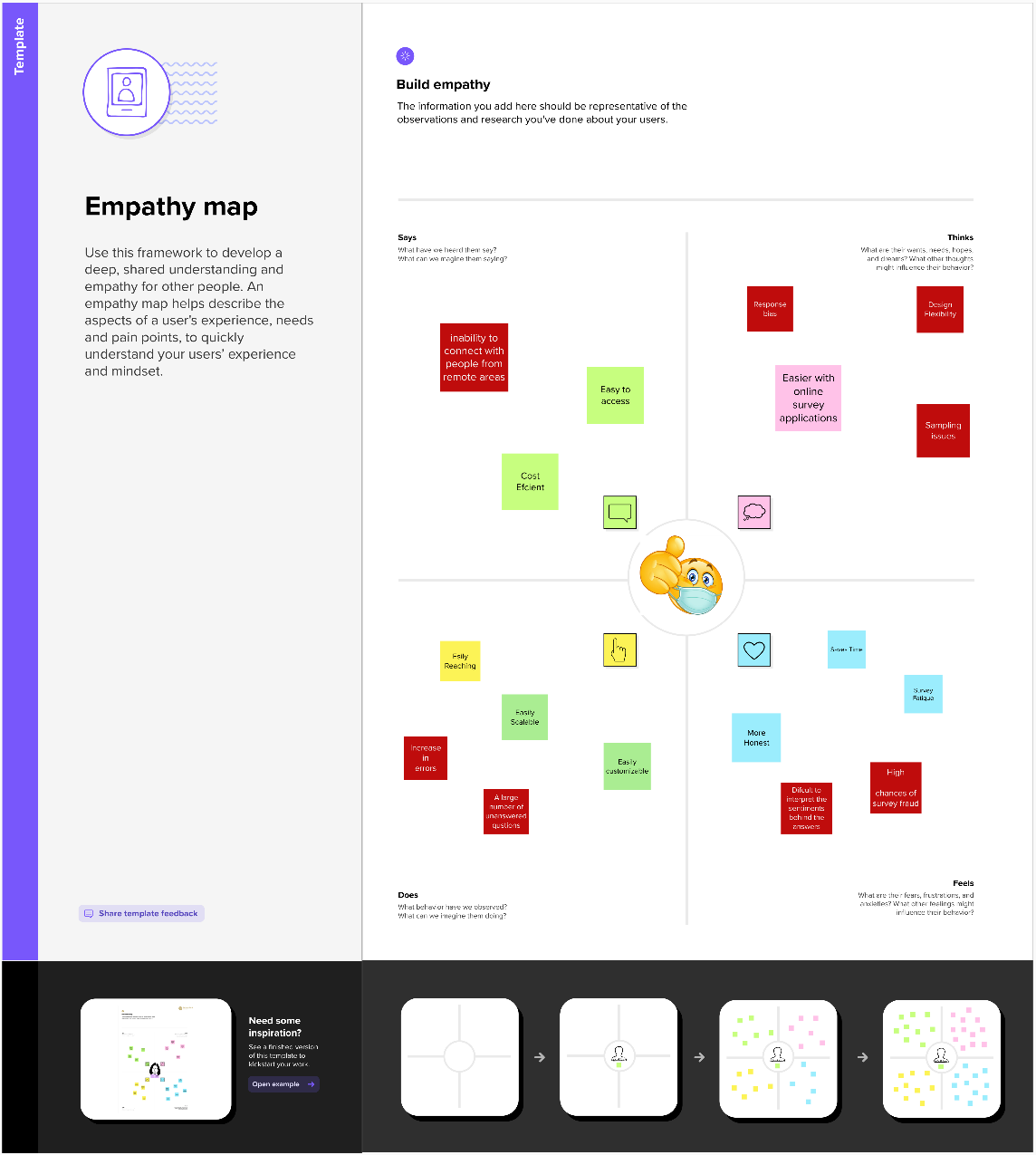
The app also provides a platform for local stores to reach a wider audience and increase their sales. By participating in Snack Squad, stores can showcase their snacks and reach new customers who might not have otherwise discovered them.

Furthermore, Snack Squad aims to promote healthy snacking habits by offering a variety of snack categories, including healthy options and dietary-restricted options. The app encourages users to make informed snack choices by providing them with nutritional information and ingredient lists.

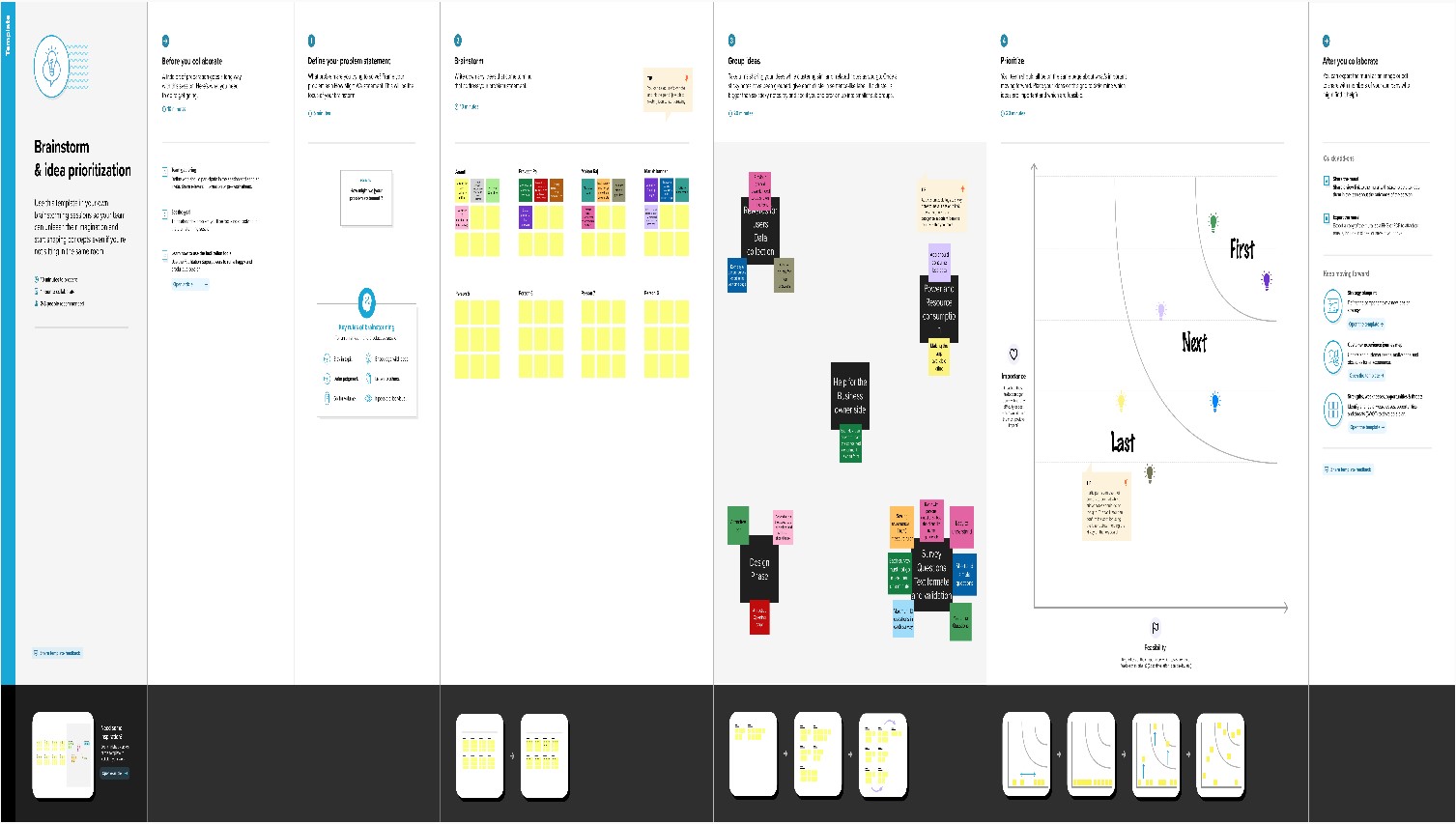
In summary, the purpose of Snack Squad is to provide a convenient, customizable, and healthy snack ordering and delivery service that benefits both users and local stores.

**Problem Definition & Design Thinking:-**

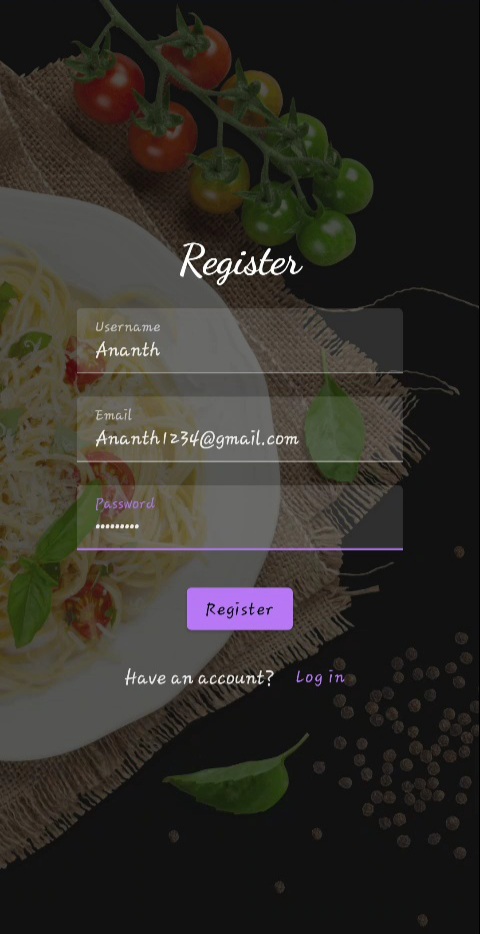
**EMPATHY MAP:**

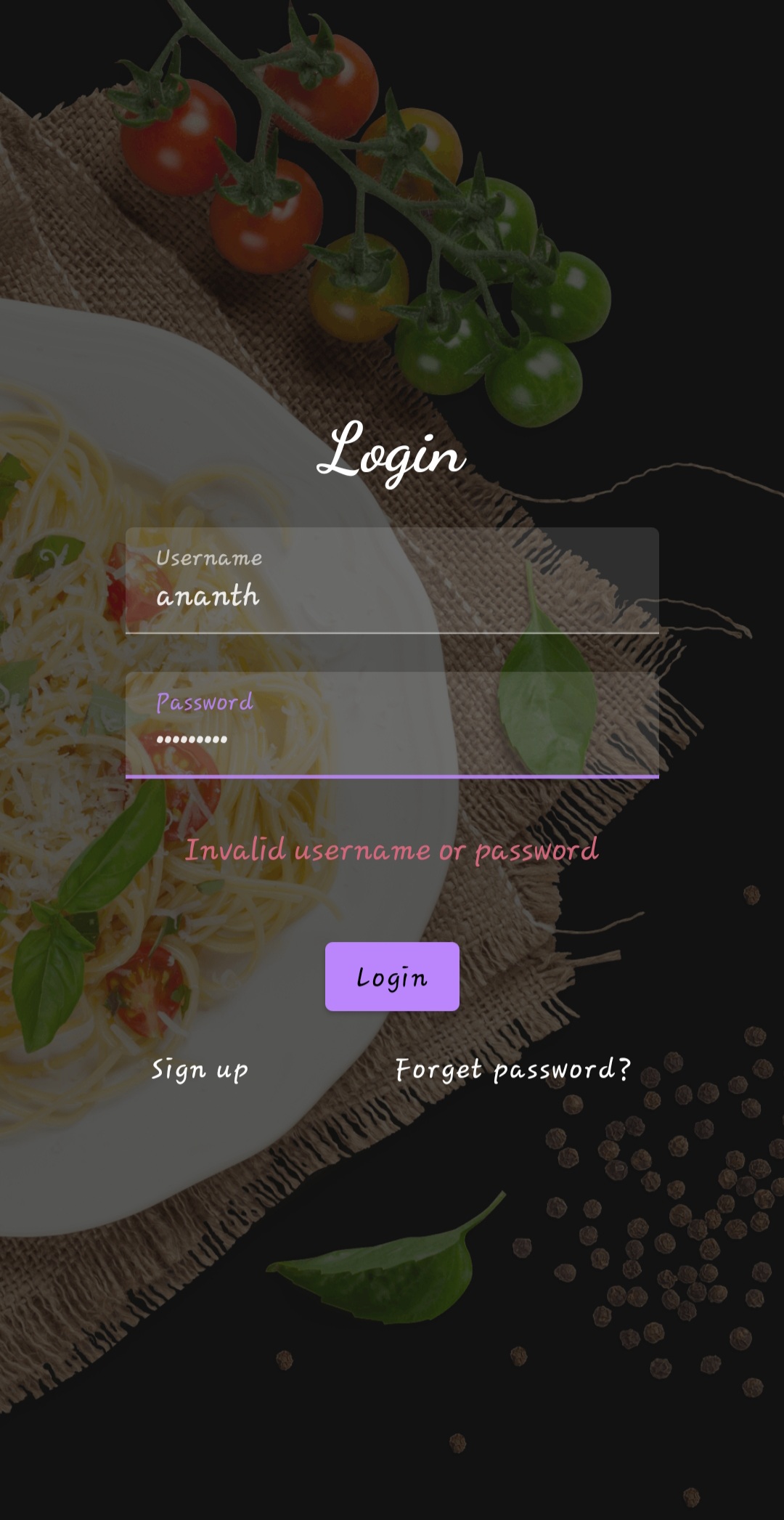
****

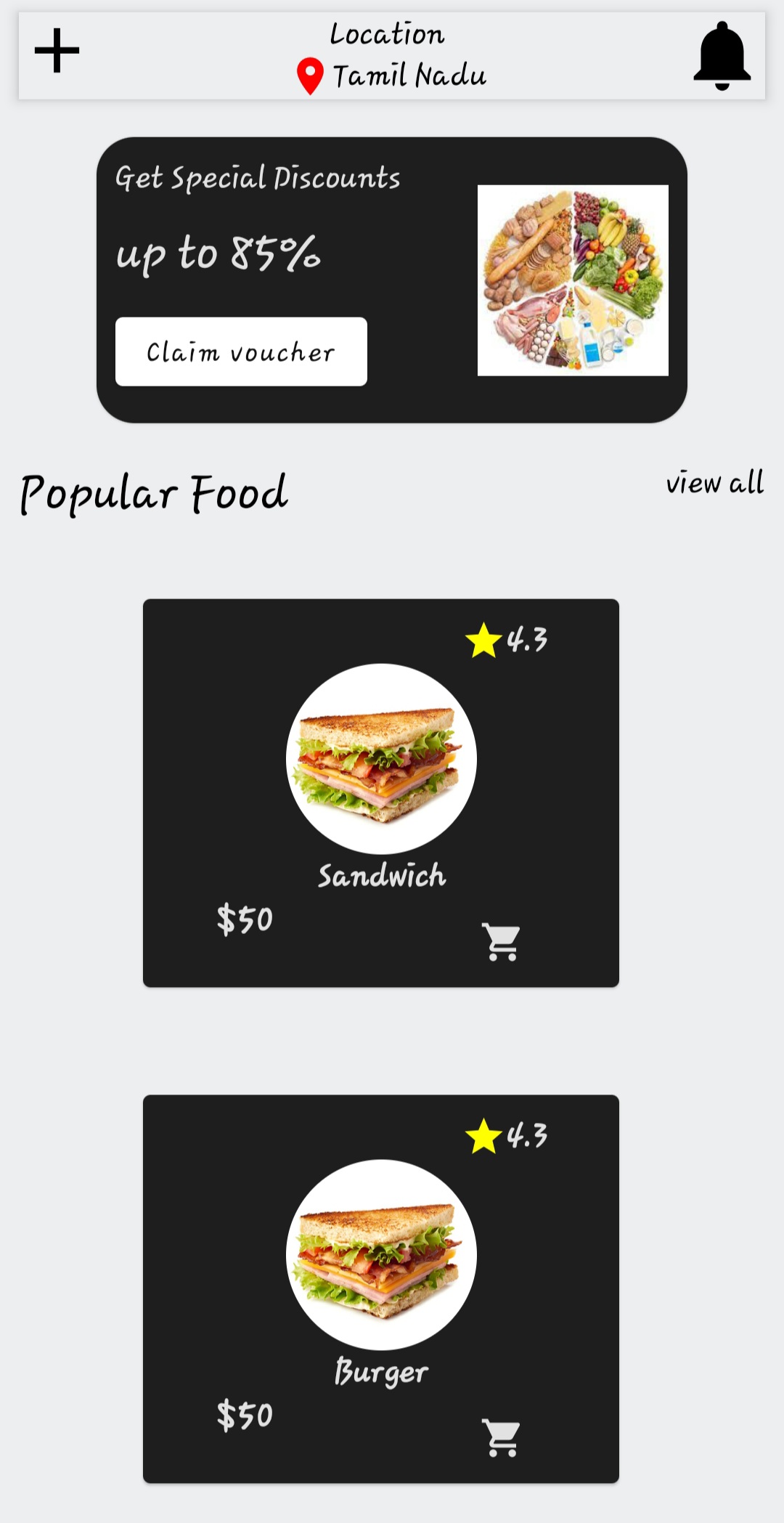
**BRAINSTORMING MAP:**

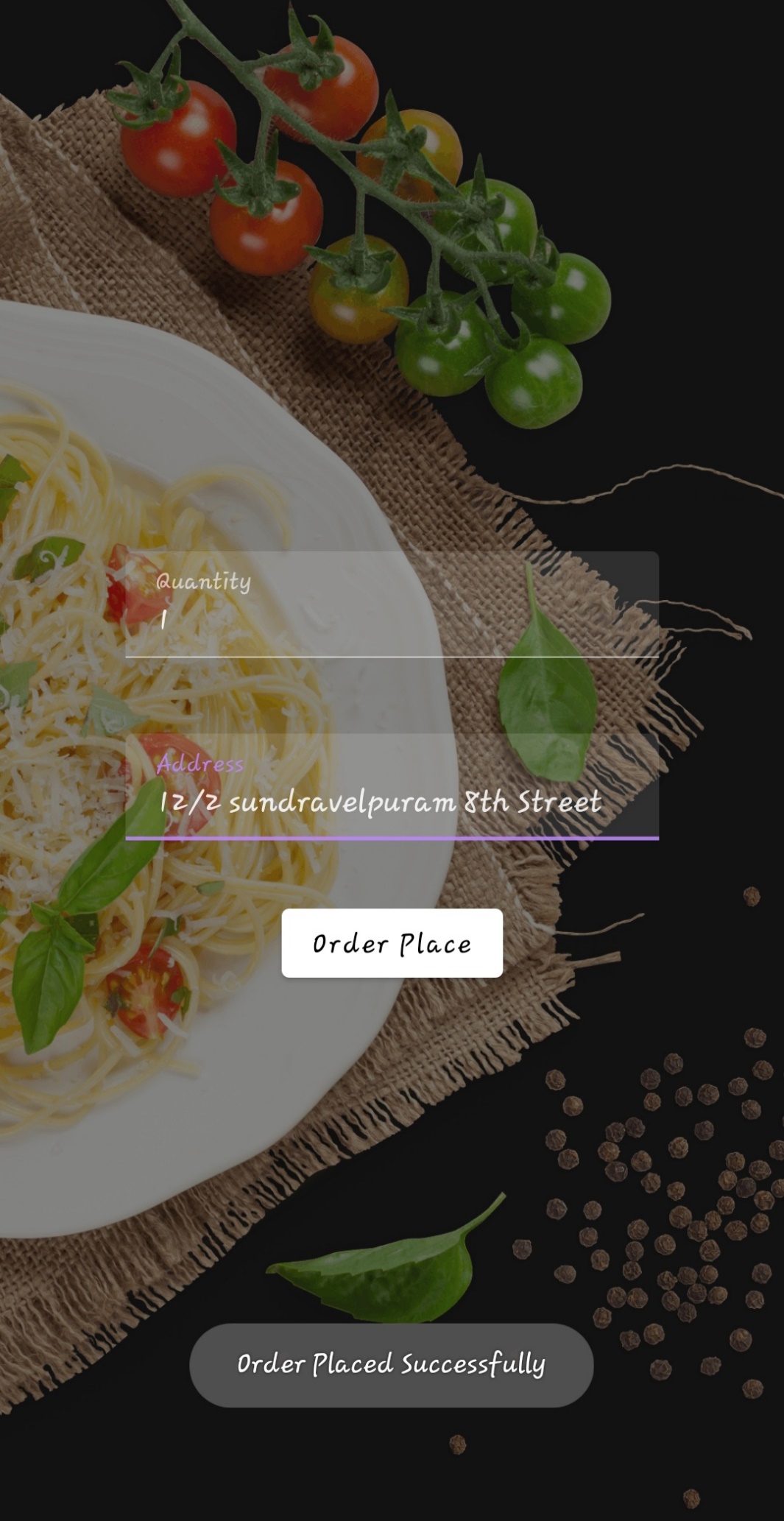
****

**RESULT:**

****

****

****

****

**ADVANTAGE:**

There are several advantages of using Snack Squad:

1. Convenience: Snack Squad allows users to order snacks from a variety of local stores and have them delivered to their doorstep, saving them time and effort.
2. Customization: The app offers a wide selection of snacks and allows users to customize their orders according to their preferences, dietary restrictions, and budgets.
3. Healthy Options: Snack Squad offers a variety of healthy snack options, which encourages users to make informed snack choices.
4. Local Support: By using Snack Squad, users support local stores and help them increase their sales.
5. Subscription Service: The app offers a subscription service, which allows users to receive regular deliveries of their favourite snacks at discounted prices.
6. Easy Reordering: The app allows users to save their favourite orders for quick and easy reordering in the future.
7. Competitive Pricing: Snack Squad offers competitive pricing, making it an affordable option for users.

Overall, Snack Squad offers a convenient, customizable, and healthy way to satisfy snack cravings while supporting local stores.

**DIS-ADVANTAGE:**

While Snack Squad offers many advantages, there are some potential disadvantages to consider:

1. Delivery Fees: Snack Squad charges delivery fees for each order, which can add up if users order frequently or for small items.
2. Limited Coverage: The app may not be available in all areas, which can limit the snack options and stores available to users.
3. Availability of Snacks: Some snack items may not be available for delivery due to stock shortages or other reasons.
4. Quality of Service: The quality of service may vary depending on the availability of delivery drivers and the accuracy of the order fulfilment process.
5. Dependence on Technology: Snack Squad is reliant on technology, and technical issues or downtime can disrupt the ordering and delivery process.
6. Packaging Waste: Snack Squad delivery may generate packaging waste, which can be detrimental to the environment.

Overall, while Snack Squad offers many advantages, it is important to consider these potential disadvantages before deciding whether to use the app.

**APPLICATION:**

Snack Squad can be applied in various settings, including:

1. Office Settings: Snack Squad can be used in office settings to provide employees with convenient and healthy snack options during work hours.
2. Residential Settings: Snack Squad can be used in residential settings to provide individuals or families with a convenient way to satisfy snack cravings without having to leave their homes.
3. Events: Snack Squad can be used for events, such as parties or meetings, to provide guests with a variety of snack options.
4. College Campuses: Snack Squad can be used on college campuses to provide students with a convenient and healthy snack option between classes.
5. Hospital Settings: Snack Squad can be used in hospital settings to provide patients with a convenient and healthy snack option during their stay.
6. Airport Lounges: Snack Squad can be used in airport lounges to provide travellers with a convenient and healthy snack option while waiting for their flights.

Overall, Snack Squad can be applied in various settings where there is a need for convenient, customizable, and healthy snack options.

**FUTURE SCOPE:**

* **Drive Sales with Social Media**

The popularity of smartphones and the usage of social media platforms using smartphones is known to the world. So, it must not be a surprise that food chains are now including provisions of using their mobile application for ordering to promote their sales.

* **Mobility and Ease**

With the online mobile payment feature ordering food using restaurant based apps has become easier these days. There occurs no requirement to make use of cash. One can order food online using online payment modes right from the restaurant ordering app.

* **Phone Orders Outstripped**

The ease and convenience of online food ordering using restaurant mobile apps make sure that Tele calling is no longer used for ordering. The reason behind this is the user-friendly interface of the food ordering app provides a smooth ordering experience to customers

* **Home Deliveries Increased**

The more the population is increasing with their increased purchasing power, the more are the situations of online food ordering occurring? Home deliveries are a matter of daily system nowadays for every restaurant, big or small in the town

**CONCLUSION:**

In conclusion, Snack Squad is a customizable snack ordering and delivery app that offers many benefits, including convenience, customization, healthy options, local support, and affordable pricing. While there are potential disadvantages to consider, such as delivery fees, limited coverage, and dependence on technology, Snack Squad can be applied in various settings to provide individuals and communities with a convenient and healthy snack option. Overall, Snack Squad addresses the problem of snack cravings while supporting local stores and promoting healthy snacking habits.

**APPENDIX**

**SOURCE CODE:**

[**https://github.com/Ananth042002/Snack-Squad.git**](https://github.com/Ananth042002/Snack-Squad.git)

**LoginActivity.kt**

package com.example.snackordering

import android.icu.text.SimpleDateFormat

import android.os.Bundle

import android.util.Log

import androidx.activity.ComponentActivity

import androidx.activity.compose.setContent

import androidx.compose.foundation.Image

import androidx.compose.foundation.layout.\*

import androidx.compose.foundation.lazy.LazyColumn

import androidx.compose.foundation.lazy.LazyRow

import androidx.compose.foundation.lazy.items

import androidx.compose.material.MaterialTheme

import androidx.compose.material.Surface

import androidx.compose.material.Text

import androidx.compose.runtime.Composable

import androidx.compose.ui.Modifier

import androidx.compose.ui.graphics.Color

import androidx.compose.ui.layout.ContentScale

import androidx.compose.ui.res.painterResource

import androidx.compose.ui.unit.dp

import androidx.compose.ui.unit.sp

import com.example.snackordering.ui.theme.SnackOrderingTheme

import java.util.\*

class AdminActivity : ComponentActivity() {

private lateinit var orderDatabaseHelper: OrderDatabaseHelper

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

orderDatabaseHelper = OrderDatabaseHelper(this)

setContent {

SnackOrderingTheme {

// A surface container using the 'background' color from the theme

Surface(

modifier = Modifier.fillMaxSize(),

color = MaterialTheme.colors.background

) {

val data=orderDatabaseHelper.getAllOrders();

Log.d("swathi" ,data.toString())

val order = orderDatabaseHelper.getAllOrders()

ListListScopeSample(order)

}

}

}

}

}

@Composable

fun ListListScopeSample(order: List<Order>) {

Image(

painterResource(id = R.drawable.order), contentDescription = "",

alpha =0.5F,

contentScale = ContentScale.FillHeight)

Text(text = "Order Tracking", modifier = Modifier.padding(top = 24.dp, start = 106.dp, bottom = 24.dp ), color = Color.White, fontSize = 30.sp)

Spacer(modifier = Modifier.height(30.dp))

LazyRow(

modifier = Modifier

.fillMaxSize()

.padding(top = 80.dp),

horizontalArrangement = Arrangement.SpaceBetween

){

item {

LazyColumn {

items(order) { order ->

Column(modifier = Modifier.padding(top = 16.dp, start = 48.dp, bottom = 20.dp)) {

Text("Quantity: ${order.quantity}")

Text("Address: ${order.address}")

}

}

}

}

}

}

**MainPage.kt**

package com.example.snackordering

import android.annotation.SuppressLint

import android.content.Context

import android.os.Bundle

import android.widget.Toast

import androidx.activity.ComponentActivity

import androidx.activity.compose.setContent

import androidx.annotation.DrawableRes

import androidx.annotation.StringRes

import androidx.compose.foundation.Image

import androidx.compose.foundation.background

import androidx.compose.foundation.layout.\*

import androidx.compose.foundation.shape.CircleShape

import androidx.compose.foundation.shape.RoundedCornerShape

import androidx.compose.material.\*

import androidx.compose.material.icons.Icons

import androidx.compose.material.icons.filled.\*

import androidx.compose.runtime.Composable

import androidx.compose.ui.Alignment

import androidx.compose.ui.Modifier

import androidx.compose.ui.draw.clip

import androidx.compose.ui.graphics.Color

import androidx.compose.foundation.lazy.LazyColumn

import androidx.compose.foundation.lazy.items

import androidx.compose.material.Text

import androidx.compose.ui.unit.dp

import androidx.compose.ui.graphics.RectangleShape

import androidx.compose.ui.layout.ContentScale

import androidx.compose.ui.platform.LocalContext

import androidx.compose.ui.res.painterResource

import androidx.compose.ui.res.stringResource

import androidx.compose.ui.text.font.FontWeight

import androidx.compose.ui.unit.sp

import androidx.core.content.ContextCompat.startActivity

import com.example.snackordering.ui.theme.SnackOrderingTheme

import android.content.Intent as Intent1

class MainPage : ComponentActivity() {

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContent {

SnackOrderingTheme {

// A surface container using the 'background' color from the theme

Surface(

modifier = Modifier.fillMaxSize(),

color = MaterialTheme.colors.background

) {

FinalView(this)

val context = LocalContext.current

//PopularFoodColumn(context)

}

}

}

}

}

@Composable

fun TopPart() {

Row(

modifier = Modifier

.fillMaxWidth()

.background(Color(0xffeceef0)), Arrangement.SpaceBetween

) {

Icon(

imageVector = Icons.Default.Add, contentDescription = "Menu Icon",

Modifier

.clip(CircleShape)

.size(40.dp),

tint = Color.Black,

)

Column(horizontalAlignment = Alignment.CenterHorizontally) {

Text(text = "Location", style = MaterialTheme.typography.subtitle1, color = Color.Black)

Row {

Icon(

imageVector = Icons.Default.LocationOn,

contentDescription = "Location",

tint = Color.Red,

)

Text(text = "Accra" , color = Color.Black)

}

}

Icon(

imageVector = Icons.Default.Notifications, contentDescription = "Notification Icon",

Modifier

.size(45.dp),

tint = Color.Black,

)

}

}

@Composable

fun CardPart() {

Card(modifier = Modifier.size(width = 310.dp, height = 150.dp), RoundedCornerShape(20.dp)) {

Row(modifier = Modifier.padding(10.dp), Arrangement.SpaceBetween) {

Column(verticalArrangement = Arrangement.spacedBy(12.dp)) {

Text(text = "Get Special Discounts")

Text(text = "up to 85%", style = MaterialTheme.typography.h5)

Button(onClick = {}, colors = ButtonDefaults.buttonColors(Color.White)) {

Text(text = "Claim voucher", color = MaterialTheme.colors.surface)

}

}

Image(

painter = painterResource(id = R.drawable.food\_tip\_im),

contentDescription = "Food Image", Modifier.size(width = 100.dp, height = 200.dp)

)

}

}

}

@Composable

fun PopularFood(

@DrawableRes drawable: Int,

@StringRes text1: Int,

context: Context

) {

Card(

modifier = Modifier

.padding(top=20.dp, bottom = 20.dp, start = 65.dp)

.width(250.dp)

) {

Column(

verticalArrangement = Arrangement.Top,

horizontalAlignment = Alignment.CenterHorizontally

) {

Spacer(modifier = Modifier.padding(vertical = 5.dp))

Row(

modifier = Modifier

.fillMaxWidth(0.7f), Arrangement.End

) {

Icon(

imageVector = Icons.Default.Star,

contentDescription = "Star Icon",

tint = Color.Yellow

)

Text(text = "4.3", fontWeight = FontWeight.Black)

}

Image(

painter = painterResource(id = drawable),

contentDescription = "Food Image",

contentScale = ContentScale.Crop,

modifier = Modifier

.size(100.dp)

.clip(CircleShape)

)

Text(text = stringResource(id = text1), fontWeight = FontWeight.Bold)

Row(modifier = Modifier.fillMaxWidth(0.7f), Arrangement.SpaceBetween) {

/\*TODO Implement Prices for each card\*/

Text(

text = "$50",

style = MaterialTheme.typography.h6,

fontWeight = FontWeight.Bold,

fontSize = 18.sp

)

IconButton(onClick = {

//var no=FoodList.lastIndex;

//Toast.

val intent = Intent1(context, TargetActivity::class.java)

context.startActivity(intent)

}) {

Icon(

imageVector = Icons.Default.ShoppingCart,

contentDescription = "shopping cart",

)

}

}

}

}

}

private val FoodList = listOf(

R.drawable.sandwish to R.string.sandwich,

R.drawable.sandwish to R.string.burgers,

R.drawable.pack to R.string.pack,

R.drawable.pasta to R.string.pasta,

R.drawable.tequila to R.string.tequila,

R.drawable.wine to R.string.wine,

R.drawable.salad to R.string.salad,

R.drawable.pop to R.string.popcorn

).map { DrawableStringPair(it.first, it.second) }

private data class DrawableStringPair(

@DrawableRes val drawable: Int,

@StringRes val text1: Int

)

@Composable

fun App(context: Context) {

Column(

modifier = Modifier

.fillMaxSize()

.background(Color(0xffeceef0))

.padding(10.dp),

verticalArrangement = Arrangement.Top,

horizontalAlignment = Alignment.CenterHorizontally

) {

Surface(modifier = Modifier, elevation = 5.dp) {

TopPart()

}

Spacer(modifier = Modifier.padding(10.dp))

CardPart()

Spacer(modifier = Modifier.padding(10.dp))

Row(modifier = Modifier.fillMaxWidth(), Arrangement.SpaceBetween) {

Text(text = "Popular Food", style = MaterialTheme.typography.h5, color = Color.Black)

Text(text = "view all", style = MaterialTheme.typography.subtitle1, color = Color.Black)

}

Spacer(modifier = Modifier.padding(10.dp))

PopularFoodColumn(context) // <- call the function with parentheses

}

}

@Composable

fun PopularFoodColumn(context: Context) {

LazyColumn(

modifier = Modifier.fillMaxSize(),

content = {

items(FoodList) { item ->

PopularFood(context = context,drawable = item.drawable, text1 = item.text1)

abstract class Context

}

},

verticalArrangement = Arrangement.spacedBy(16.dp))

}

@SuppressLint("UnusedMaterialScaffoldPaddingParameter")

@Composable

fun FinalView(mainPage: MainPage) {

SnackOrderingTheme {

Scaffold() {

val context = LocalContext.current

App(context)

}

}}

**RegisterActivity.kt**

package com.example.snackordering

import android.content.Context

import android.content.Intent

import android.os.Bundle

import androidx.activity.ComponentActivity

import androidx.activity.compose.setContent

import androidx.compose.foundation.Image

import androidx.compose.foundation.layout.\*

import androidx.compose.material.\*

import androidx.compose.runtime.\*

import androidx.compose.ui.Alignment

import androidx.compose.ui.Modifier

import androidx.compose.ui.graphics.Color

import androidx.compose.ui.layout.ContentScale

import androidx.compose.ui.res.painterResource

import androidx.compose.ui.text.font.FontFamily

import androidx.compose.ui.text.font.FontWeight

import androidx.compose.ui.unit.dp

import androidx.compose.ui.unit.sp

import androidx.core.content.ContextCompat

import com.example.snackordering.ui.theme.SnackOrderingTheme

class MainActivity : ComponentActivity() {

private lateinit var databaseHelper: UserDatabaseHelper

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

databaseHelper = UserDatabaseHelper(this)

setContent {

SnackOrderingTheme {

// A surface container using the 'background' color from the theme

Surface(

modifier = Modifier.fillMaxSize(),

color = MaterialTheme.colors.background

) {

RegistrationScreen(this,databaseHelper)

}

}

}

}

}

@Composable

fun RegistrationScreen(context: Context, databaseHelper: UserDatabaseHelper) {

Image(

painterResource(id = R.drawable.order), contentDescription = "",

alpha =0.3F,

contentScale = ContentScale.FillHeight,

)

var username by remember { mutableStateOf("") }

var password by remember { mutableStateOf("") }

var email by remember { mutableStateOf("") }

var error by remember { mutableStateOf("") }

Column(

modifier = Modifier.fillMaxSize(),

horizontalAlignment = Alignment.CenterHorizontally,

verticalArrangement = Arrangement.Center

) {

Text(

fontSize = 36.sp,

fontWeight = FontWeight.ExtraBold,

fontFamily = FontFamily.Cursive,

color = Color.White,

text = "Register"

)

Spacer(modifier = Modifier.height(10.dp))

TextField(

value = username,

onValueChange = { username = it },

label = { Text("Username") },

modifier = Modifier

.padding(10.dp)

.width(280.dp)

)

TextField(

value = email,

onValueChange = { email = it },

label = { Text("Email") },

modifier = Modifier

.padding(10.dp)

.width(280.dp)

)

TextField(

value = password,

onValueChange = { password = it },

label = { Text("Password") },

modifier = Modifier

.padding(10.dp)

.width(280.dp)

)

if (error.isNotEmpty()) {

Text(

text = error,

color = MaterialTheme.colors.error,

modifier = Modifier.padding(vertical = 16.dp)

)

}

Button(

onClick = {

if (username.isNotEmpty() && password.isNotEmpty() && email.isNotEmpty()) {

val user = User(

id = null,

firstName = username,

lastName = null,

email = email,

password = password

)

databaseHelper.insertUser(user)

error = "User registered successfully"

// Start LoginActivity using the current context

context.startActivity(

Intent(

context,

LoginActivity::class.java

)

)

} else {

error = "Please fill all fields"

}

},

modifier = Modifier.padding(top = 16.dp)

) {

Text(text = "Register")

}

Spacer(modifier = Modifier.width(10.dp))

Spacer(modifier = Modifier.height(10.dp))

Row() {

Text(

modifier = Modifier.padding(top = 14.dp), text = "Have an account?"

)

TextButton(onClick = {

context.startActivity(

Intent(

context,

LoginActivity::class.java

)

)

})

{

Spacer(modifier = Modifier.width(10.dp))

Text(text = "Log in")

}

}

}

}

private fun startLoginActivity(context: Context) {

val intent = Intent(context, LoginActivity::class.java)

ContextCompat.startActivity(context, intent, null)

}

**TargetActivity.kt**

package com.example.snackordering

import android.content.Context

import android.content.Intent

import android.os.Bundle

import android.util.Log

import android.widget.Toast

import androidx.activity.ComponentActivity

import androidx.activity.compose.setContent

import androidx.compose.foundation.Image

import androidx.compose.foundation.background

import androidx.compose.foundation.layout.\*

import androidx.compose.foundation.text.KeyboardActions

import androidx.compose.foundation.text.KeyboardOptions

import androidx.compose.material.\*

import androidx.compose.runtime.\*

import androidx.compose.ui.Alignment

import androidx.compose.ui.Modifier

import androidx.compose.ui.graphics.Color

import androidx.compose.ui.layout.ContentScale

import androidx.compose.ui.platform.LocalContext

import androidx.compose.ui.platform.textInputServiceFactory

import androidx.compose.ui.res.painterResource

import androidx.compose.ui.text.input.KeyboardType

import androidx.compose.ui.tooling.preview.Preview

import androidx.compose.ui.unit.dp

import androidx.core.content.ContextCompat

import com.example.snackordering.ui.theme.SnackOrderingTheme

class TargetActivity : ComponentActivity() {

private lateinit var orderDatabaseHelper: OrderDatabaseHelper

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

orderDatabaseHelper = OrderDatabaseHelper(this)

setContent {

SnackOrderingTheme {

// A surface container using the 'background' color from the theme

Surface(

modifier = Modifier

.fillMaxSize()

.background(Color.White)

) {

Order(this, orderDatabaseHelper)

val orders = orderDatabaseHelper.getAllOrders()

Log.d("swathi", orders.toString())

}

}

}

}

}

@Composable

fun Order(context: Context, orderDatabaseHelper: OrderDatabaseHelper){

Image(painterResource(id = R.drawable.order), contentDescription = "",

alpha =0.5F,

contentScale = ContentScale.FillHeight)

Column(

horizontalAlignment = Alignment.CenterHorizontally,

verticalArrangement = Arrangement.Center) {

val mContext = LocalContext.current

var quantity by remember { mutableStateOf("") }

var address by remember { mutableStateOf("") }

var error by remember { mutableStateOf("") }

TextField(value = quantity, onValueChange = {quantity=it},

label = { Text("Quantity") },

keyboardOptions = KeyboardOptions(keyboardType = KeyboardType.Number),

modifier = Modifier

.padding(10.dp)

.width(280.dp))

Spacer(modifier = Modifier.padding(10.dp))

TextField(value = address, onValueChange = {address=it},

label = { Text("Address") },

modifier = Modifier

.padding(10.dp)

.width(280.dp))

Spacer(modifier = Modifier.padding(10.dp))

if (error.isNotEmpty()) {

Text(

text = error,

color = MaterialTheme.colors.error,

modifier = Modifier.padding(vertical = 16.dp)

)

}

Button(onClick = {

if( quantity.isNotEmpty() and address.isNotEmpty()){

val order = Order(

id = null,

quantity = quantity,

address = address

)

orderDatabaseHelper.insertOrder(order)

Toast.makeText(mContext, "Order Placed Successfully", Toast.LENGTH\_SHORT).show()}

},

colors = ButtonDefaults.buttonColors(backgroundColor = Color.White))

{

Text(text = "Order Place", color = Color.Black)

}

}

}

private fun startMainPage(context: Context) {

val intent = Intent(context, LoginActivity::class.java)

ContextCompat.startActivity(context, intent, null)