A Personalized Travel Planning and Tracking App

1.INTRODUCTION

Overview

A personalized travel planning and tracking app developed in Android Studio is a mobile application that helps travellers plan and track their trips according to their preferences and needs. The app is designed to provide users with a personalized and seamless travel experience by offering a range of features that enable them to plan, book, and manage their trips all in one place.

The app enables users to plan their trips by setting up travel itineraries, selecting destinations, and creating a schedule. Users can also add details such as flights, accommodation, activities, and transportation to their itinerary. The app allows users to book their flights, hotels, and activities directly from the app. Users can also make payments through the app, making it a convenient and secure way to book and manage travel arrangements.

The app provides users with real-time updates on their travel plans, including flight delays, cancellations, and gate changes. Users can also track their luggage and receive alerts if their bags are delayed or misplaced. The app can be personalized according to users' preferences and needs. For example, users can customize their itineraries based on their budget, interests, and travel style.

The app enables users to share their travel plans and experiences with friends and family on social media. Users can also connect with other travellers and share tips and recommendations.

Purpose

The purpose of a personalized travel planning and tracking app developed in Android Studio is to provide users with a convenient and personalized tool for planning, managing, and tracking their travels. The app is designed to help users streamline the travel planning process and ensure a smooth and stress-free travel experience. The app enables users to plan their trips quickly and efficiently by providing tools and resources for itinerary planning, flight and hotel booking, transportation, and activity scheduling.

The app allows users to customize their travel plans based on their preferences and interests, ensuring that they have a personalized and enjoyable travel experience. The app provides users with real-time updates on their travel plans, including flight delays, cancellations, gate changes, and baggage tracking.

The app provides a secure platform for users to book their flights, hotels, and activities and make payments directly through the app, minimizing the risk of fraud and ensuring a hassle-free booking process.

The app allows users to track their travel plans, including flight and hotel reservations, transportation, and activities, providing a centralized and easy-to-access source of information.

2.Problem Definition & Design Thinking

2.1 Empathy Map

Template

Empathy map

Use this framework to develop a deep, shared understanding and empathy for other people. An empathy map helps describe the aspects of a user's experience, needs and pain points, to quickly understand your users' experience and mindset.

Share template feedback

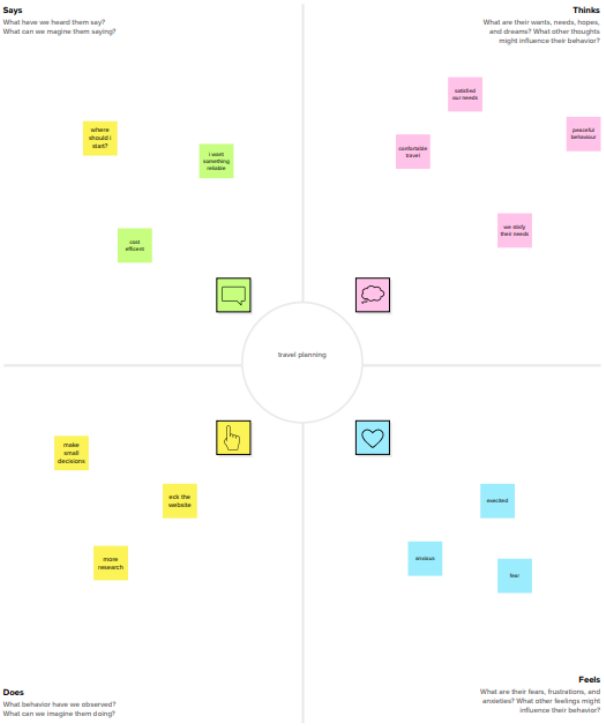
Need some inspiration?

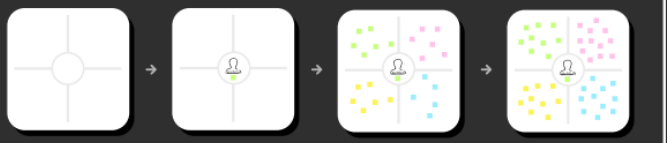
See a finished version of this template to kickstart your work.

Open example →

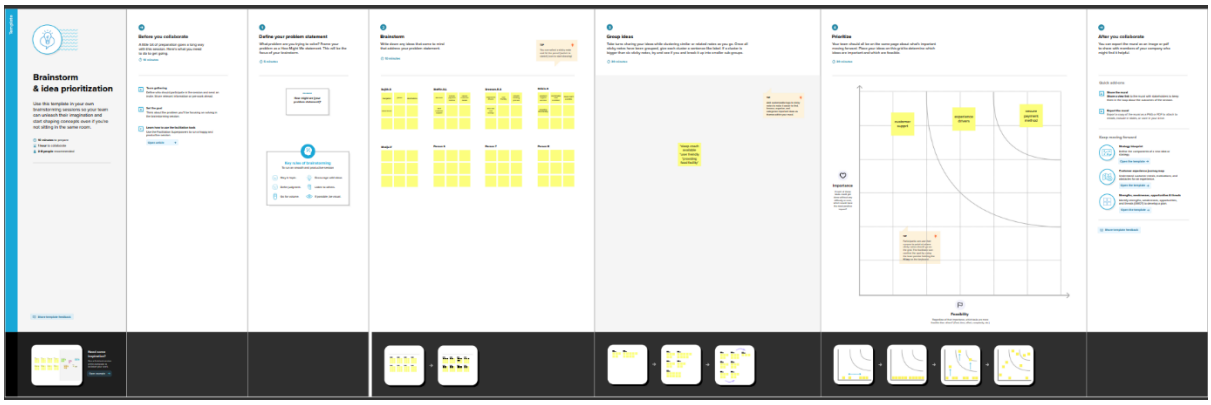
Build empathy

The information you add here should be representative of the observations and research you've done about your users.





2.2 Ideation and Brainstorming Map



RESULT

Register

Username
pradheesh

Email
pradheeshpv03@gmail.com

Password
•••••

Register

Have an account? Log in

Login

Username

pradheesh

Password

•••••

Login

Register

Forget password?

10:13

Wanderlust Travel

Bali

Super saver pack with less than $10000

7days/2persons

Paris

Super saver pack with less than $10000

7days/2persons

Bali

Super saver pack with less than $10000

7days/2persons

Paris

Super saver pack with less than $10000

7days/2persons

Singapore

Super saver pack with less than $10000

7days/2persons

Bali

Day 1: Arrival and Relaxation
Arrive in Bali and check into your hotel or accommodation.
Spend the day relaxing and getting acclimated to the island.
If you have time, explore the nearby area or head to the beach.

Day 2: Ubud Tour
Start your day early and head to Ubud, a cultural and artistic hub in Bali.
Visit the Monkey Forest and the Ubud Palace.
Take a tour of the Tegalalang Rice Terrace, a beautiful UNESCO World Heritage Site.
End your day with a traditional Balinese dance performance.

Day 3: Temple Hopping
Visit some of Bali's most famous temples, such as Tanah Lot and Uluwatu.

Paris

Day 1: Arrival and Introduction
Check into your accommodation and freshen up
Take a stroll around the neighborhood to get acquainted
Visit the Eiffel Tower, preferably in the evening when it is lit up
Have a relaxing dinner at a nearby restaurant

Day 2: Art and History
Visit the Louvre Museum to see some of the world's most famous art pieces
Stroll through the Tuileries Garden and the Place de la Concorde
Visit the Orsay Museum, which houses a large collection of impressionist art
Have dinner at a local French restaurant

Day 3: French Culture and Food
Visit the Montmartre neighborhood to see the famous Basilique du Sacré-Cœur and

Singapore

Day 1:
Morning: Visit Gardens by the Bay and marvel at the Supertree Grove and the Flower Dome and Cloud Forest conservatories.
Afternoon: Explore the Marina Bay Sands complex, which includes a casino, luxury shopping mall, and observation deck with a stunning view of the city.

Day 2:
Morning: Explore the historic district of Chinatown, including the Buddha Tooth Relic Temple and Museum and the Sri Mariamman Temple.
Afternoon: Visit the nearby Clarke Quay for lunch and to explore its waterfront restaurants, bars, and shops.

Day 3:
Morning: Take a tour of the UNESCO-listed

ADVANTAGES & DISADVANTAGES

Advantages

- Customized travel plans: The app can help users create personalized travel plans based on their preferences, interests, and budget. It can suggest destinations, flights, hotels, and activities, and help users make informed decisions about their trip.

- Real-time updates: The app can provide real-time information about flights, weather, traffic, and other factors that can affect travel plans. Users can receive alerts and notifications about delays, cancellations, or other changes, and adjust their itinerary accordingly.

- Interactive maps: The app can offer interactive maps that allow users to explore their destination, find nearby attractions, and plan their routes. It can also provide directions and recommendations for restaurants, shops, and other places of interest.

- Social sharing: The app can allow users to share their travel experiences with friends and family through social media. It can also provide an opportunity for users to connect with other travellers and exchange tips and advice.

Disadvantages

- Limited Device Compatibility: The app developed in Android Studio may not be compatible with all devices, limiting the number of people who can use the app. This may also affect the user experience of those who are able to use it if the app does not function properly on their device.

- Technical Issues: Any app developed in Android Studio may have technical issues such as bugs, crashes, and slow loading times. These issues can impact the user experience and make the app frustrating to use.

- Privacy Concerns: The app may require users to provide personal information such as their location, travel itinerary, and personal preferences in order to provide personalized recommendations. This may raise privacy concerns for some users who are not comfortable sharing this information.

- Dependence on Internet Connection: The app may require an internet connection to function properly, which could be problematic if the user is traveling to areas with limited or no internet access.

- Lack of Personal Touch: While the app can provide personalized recommendations based on the user's preferences and travel history, it may lack the personal touch and human interaction that some travellers prefer when planning their trips.

APPLICATIONS

- Customized Travel Planning: With a personalized travel planning app, users can create custom travel itineraries that cater to their preferences and interests. Users can input details about their budget, preferred activities, and travel dates, and the app will provide recommendations for accommodations, attractions, restaurants, and other activities that suit their interests.

- Real-time Tracking: Travelers can use the app to track their itinerary, monitor flight or train schedules, and receive real-time updates about changes or delays. This can help them stay on schedule and make necessary adjustments on the go.

- Location-based Recommendations: The app can use location data to provide personalized recommendations for nearby attractions, restaurants, and events. This feature can help travellers discover new places and experiences they may not have otherwise known about.

- Social Sharing: Users can share their travel plans, experiences, and recommendations with friends and family through social media or within the app. This can help build a community of travellers and provide useful insights and tips for others planning similar trips.

CONCLUSION

In conclusion, a Personalized Travel Planning and Tracking App developed in android studio can offer a wide range of benefits for both travellers and tourism industry professionals. With features such as customized travel planning, real-time tracking, location-based recommendations, social sharing, and analytics, the app can help users create personalized itineraries that cater to their interests and preferences, while also providing valuable insights for tourism industry professionals. As more people turn to technology for travel planning and organization, the development of such apps is likely to become increasingly important in the tourism industry.

FUTURE SCOPE

- Artificial Intelligence and Machine Learning: Integrating AI and machine learning capabilities into the app could allow it to learn from user behaviour and provide even more personalized recommendations and travel plans.

- Virtual Reality and Augmented Reality: Incorporating VR and AR technologies into the app could allow users to experience destinations and attractions virtually before they arrive, helping them make more informed travel decisions.

- Blockchain Technology: Implementing blockchain technology could help enhance the security and privacy of user data and transactions, which is especially important in the travel industry.

- Smart City Integration: Integrating with smart city technologies could allow the app to provide even more detailed and accurate recommendations based on real-time data about traffic, weather, and other factors.

- Sustainability and Responsible Tourism: As more travellers become conscious of the impact of their travel on the environment and local communities, the app could integrate features that promote sustainable and responsible tourism practices, such as eco-friendly accommodations and tours.

Overall, the future scope of a Personalized Travel Planning and Tracking App developed in android studio is vast, and there are many opportunities to continue enhancing and improving the app to meet the evolving needs of travellers and the tourism industry.

Appendix

A. Source code

AndroidManifest.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="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/Theme.TravelApp"
        tools:targetApi="31">
        <activity
            android:name=".RegisterActivity"
            android:exported="false"
            android:label="RegisterActivity"
            android:theme="@style/Theme.TravelApp" />
        <activity
            android:name=".SingaporeActivity"
            android:exported="false"
```

```
    android:label="@string/title_activity_singapore"
    android:theme="@style/Theme.TravelApp" />
<activity
    android:name=".ParisActivity"
    android:exported="false"
    android:label="@string/title_activity_paris"
    android:theme="@style/Theme.TravelApp" />
<activity
    android:name=".BaliActivity"
    android:exported="false"
    android:label="@string/title_activity_bali"
    android:theme="@style/Theme.TravelApp" />
<activity
    android:name=".MainActivity"
    android:exported="true"
    android:label="@string/app_name"
    android:theme="@style/Theme.TravelApp"/>
<activity
    android:name=".LoginActivity"
    android:exported="true"
    android:label="@string/app_name"
    android:theme="@style/Theme.TravelApp">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
```

```
            <category
android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
  </application>

</manifest>
```

BaliActivity.kt

```
package com.example.travelapp

import android.os.Bundle
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.rememberScrollState
import androidx.compose.foundation.verticalScroll
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.draw.scale
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.res.stringResource
import androidx.compose.ui.text.font.FontFamily
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import com.example.travelapp.ui.theme.TravelAppTheme

class BaliActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            TravelAppTheme {
                // A surface container using the 'background' color from the
theme
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colors.background
                ) {
                    PlaceOne()
                }
            }
        }
    }
```

```
}

@Composable
fun PlaceOne() {
    Column(modifier = Modifier.background(color = Color.White)
        .padding(20.dp)
        .verticalScroll(rememberScrollState())
    ) {
        Text(
            fontSize = 40.sp,
            color = Color(android.graphics.Color.rgb(120, 40, 251)),
            fontFamily = FontFamily.Cursive,
            text = stringResource(id = R.string.place_1),
        )
        Image(
            painterResource(id = R.drawable.bali), contentDescription =
"",
            modifier = Modifier
                .padding(16.dp)
                .fillMaxWidth()
                .height(200.dp)
                .scale(scaleX = 1.2F, scaleY = 1F)
        )
        Text(
            color=Color.Black,
            text = "Day 1: Arrival and Relaxation\n" +
```

"Arrive in Bali and check into your hotel or accommodation.\n" +

"Spend the day relaxing and getting acclimated to the island.\n" +

"If you have time, explore the nearby area or head to the beach.\n" +

"\n" +

"Day 2: Ubud Tour\n" +

"Start your day early and head to Ubud, a cultural and artistic hub in Bali.\n" +

"Visit the Monkey Forest and the Ubud Palace.\n" +

"Take a tour of the Tegalalang Rice Terrace, a beautiful UNESCO World Heritage Site.\n" +

"End your day with a traditional Balinese dance performance.\n" +

"\n" +

"Day 3: Temple Hopping\n" +

"Visit some of Bali's most famous temples, such as Tanah Lot and Uluwatu.\n" +

"Take in the stunning views of the ocean and cliffs.\n" +

"Enjoy a sunset dinner at one of the many restaurants near the temples.\n" +

"\n" +

"Day 4: Waterfalls and Beaches\n" +

```
                "Take a day trip to Bali's beautiful waterfalls, such as Tegenungan or Gitgit.\n" +
                "Spend the afternoon at one of Bali's world-renowned beaches, like Seminyak or Nusa Dua.\n" +
                "\n" +
                "Day 5: Island Hopping\n" +


                "Take a day trip to one of Bali's neighboring islands, such as Nusa Lembongan or Gili Islands.\n" +
                "Snorkel or scuba dive in the clear waters and relax on the beach.\n" +
                "\n" +
                "Day 6: Cultural Activities\n" +
                "Visit a traditional Balinese village and learn about the island.\n" +
            "\n" +
                "Day 7: Departure\n" +
                "Explore the surrounding area and take in the stunning sunset views.\n" +
                "Have dinner at a local restaurant before returning to your accommodation."
        )



    }
}
```

LoginActivity.kt

```
package com.example.travelapp

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.background
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.text.input.PasswordVisualTransformation
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
```

```
import androidx.core.content.ContextCompat

class LoginActivity : ComponentActivity() {
    private lateinit var databaseHelper: UserDatabaseHelper
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        databaseHelper = UserDatabaseHelper(this)
        setContent {
            LoginScreen(this, databaseHelper)
        }
    }
}
@Composable
fun LoginScreen(context: Context, databaseHelper:
UserDatabaseHelper) {

    var username by remember { mutableStateOf("") }
    var password by remember { mutableStateOf("") }
    var error by remember { mutableStateOf("") }

    Column(
        modifier = Modifier.fillMaxSize().background(Color.White),
        horizontalAlignment = Alignment.CenterHorizontally,
        verticalArrangement = Arrangement.Center
    ) {
```

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


    Text(
        fontSize = 36.sp,
        fontWeight = FontWeight.ExtraBold,
        fontFamily = FontFamily.Cursive,
        text = "Login"
    )
    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 = password,
        onValueChange = { password = it },
        label = { Text("Password") },
        visualTransformation = PasswordVisualTransformation(),
        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()) {
                    val user =
databaseHelper.getUserByUsername(username)
                    if (user != null && user.password == password) {
                        error = "Successfully log in"
                        context.startActivity(
                            Intent(
                                context,
                                MainActivity::class.java
                            )
                        )
                        //onLoginSuccess()
                    }
                    else {
```

```
                error =  "Invalid username or password"
            }

        } else {
            error = "Please fill all fields"
        }
    },
    modifier = Modifier.padding(top = 16.dp)
) {
    Text(text = "Login")
}
Row {
    TextButton(onClick = {context.startActivity(
        Intent(
            context,
            RegisterActivity::class.java
        )
    )}
    )
    { Text(text = "Register") }
    TextButton(onClick = {
    })

    {
        Spacer(modifier = Modifier.width(60.dp))
```

```
            Text(text = "Forget password?")
        }
    }
  }
}
private fun startMainPage(context: Context) {
  val intent = Intent(context, MainActivity::class.java)
  ContextCompat.startActivity(context, intent, null)
}
```

MainActivity.kt

```
package com.example.travelapp

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.clickable
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.verticalScroll
import androidx.compose.material.Card
import androidx.compose.material.Text
```

```
import androidx.compose.runtime.Composable
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.draw.scale
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.res.stringResource
import androidx.compose.ui.text.font.FontFamily
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.text.style.TextAlign
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp

class MainActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            TravelApp(this)
        }
    }


    @Composable
    fun TravelApp(context: Context) {
        Column(
```

```
    modifier = Modifier
        .padding(20.dp)
        .verticalScroll(rememberScrollState())

) {

    Text(
        fontSize = 40.sp,
        color = Color(android.graphics.Color.rgb(120, 40, 251)),
        fontFamily = FontFamily.Cursive,
        text = "Wanderlust Travel"
    )

    Spacer(modifier = Modifier.height(20.dp))

    // 01
    Card(
        modifier = Modifier
            .fillMaxWidth()
            .height(250.dp)
            .clickable {
            context.startActivity(
                Intent(context, BaliActivity::class.java)

            )
```

```
            },
            elevation = 8.dp
        )
        {
            Column(
                horizontalAlignment = Alignment.CenterHorizontally
            ) {
                Image(
                    painterResource(id = R.drawable.bali),
contentDescription = "",
                    modifier = Modifier
                        .height(150.dp)
                        .scale(scaleX = 1.2F, scaleY = 1F)
                )

                Text(
                    text = stringResource(id = R.string.place_1),
                    fontSize = 18.sp
                )


                Text(
                    text = stringResource(id = R.string.description),
                    fontWeight = FontWeight.Light,
                    fontSize = 16.sp,
                    textAlign = TextAlign.Center,
```

```
            )

            Text(
                text = stringResource(id = R.string.plan), color =
Color.Gray,
                fontSize = 16.sp
            )
        }
    }

    Spacer(modifier = Modifier.height(20.dp))



    //02
    Card(
        modifier = Modifier
            .fillMaxWidth()
            .height(250.dp)
            .clickable {
            context.startActivity(
                Intent(context, ParisActivity::class.java)

            )
            },
        elevation = 8.dp
    )
```

```
        {
            Column(
                horizontalAlignment = Alignment.CenterHorizontally
            ) {
                Image(
                    painterResource(id = R.drawable.paris),
contentDescription = "",
                    modifier = Modifier
                        .height(150.dp)
                        .scale(scaleX = 1.2F, scaleY = 1F)
                )

                Text(
                    text = stringResource(id = R.string.place_2),
                    fontSize = 18.sp
                )



                Text(
                    text = stringResource(id = R.string.description),
                    fontWeight = FontWeight.Light,
                    fontSize = 16.sp,
                    textAlign = TextAlign.Center,
                )

                Text(
```

```
                text = stringResource(id = R.string.plan), color =
Color.Gray,
                fontSize = 16.sp
            )
        }
    }

    Spacer(modifier = Modifier.height(20.dp))

    //03
    Card(
        modifier = Modifier
            .fillMaxWidth()
            .height(250.dp)
            .clickable {
            context.startActivity(
                Intent(context, SingaporeActivity::class.java)

            )
            },
        elevation = 8.dp
    )
    {
        Column(
            horizontalAlignment = Alignment.CenterHorizontally
        ) {
```

```
            Image(
                painterResource(id = R.drawable.singapore),
contentDescription = "",
                modifier = Modifier
                    .height(150.dp)
                    .scale(scaleX = 1.2F, scaleY = 1F)
            )

            Text(
                text = stringResource(id = R.string.place_3),
                fontSize = 18.sp
            )



            Text(
                text = stringResource(id = R.string.description),
                fontWeight = FontWeight.Light,
                fontSize = 16.sp,
                textAlign = TextAlign.Center,
            )

            Text(
                text = stringResource(id = R.string.plan), color =
Color.Gray,
                fontSize = 16.sp
            )
```

```
            }
        }

        Spacer(modifier = Modifier.height(20.dp))
    }
  }
}
```

ParisActivity.kt

```
package com.example.travelapp

import android.os.Bundle
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.rememberScrollState
import androidx.compose.foundation.verticalScroll
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.draw.scale
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.res.stringResource
import androidx.compose.ui.text.font.FontFamily
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import com.example.travelapp.ui.theme.TravelAppTheme

class ParisActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            TravelAppTheme {
                // A surface container using the 'background' color from the
theme
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colors.background
                ) {
                    Greeting()
                }
            }
        }
    }
```

```
}

@Composable
fun Greeting() {

    Column(
        modifier = Modifier.background(color = Color.White)
            .padding(20.dp)
            .verticalScroll(rememberScrollState())
    ) {
        Text(
            fontSize = 40.sp,
            color = Color(android.graphics.Color.rgb(120, 40, 251)),
            fontFamily = FontFamily.Cursive,
            text = stringResource(id = R.string.place_2),
        )
        Image(
            painterResource(id = R.drawable.paris), contentDescription =
"",
            modifier = Modifier
                .padding(16.dp)
                .fillMaxWidth()
                .height(200.dp)
                .scale(scaleX = 1.2F, scaleY = 1F)
        )
        Text(
```

```
color=Color.Black,
text = "Day 1: Arrival and Introduction\n" +

"Check into your accommodation and freshen up\n" +
"Take a stroll around the neighborhood to get acquainted\n" +
"Visit the Eiffel Tower, preferably in the evening when it is lit up\n" +
"Have a relaxing dinner at a nearby restaurant\n" +

"\n" +
"Day 2: Art and History\n" +

"Visit the Louvre Museum to see some of the world's most famous art pieces\n" +
"Stroll through the Tuileries Garden and the Place de la Concorde\n" +
"Visit the Orsay Museum, which houses a large collection of impressionist art\n" +
"Have dinner at a local French restaurant\n" +

"\n" +
"Day 3: French Culture and Food\n" +

"Visit the Montmartre neighborhood to see the famous Basilique du Sacré-Cœur and Place du Tertre\n" +
"Explore the historic neighborhood of Le Marais\n" +
```

```
"Try some delicious French pastries at a local bakery\n" +
"Have dinner at a brasserie to taste some classic French cuisine\n" +

"\n" +

"Day 4: Architecture and Gardens\n" +

"Visit the Palace of Versailles, a UNESCO World Heritage site, and explore its beautiful gardens\n" +
"Walk along the Champs-Elysées and stop at the Arc de Triomphe\n" +
"Visit the Sainte-Chapelle, a beautiful Gothic chapel with stunning stained-glass windows\n" +
"Have dinner at a local restaurant in the 7th arrondissement\n" +

"\n" +
"Day 5: Shopping and Sightseeing\n" +

"Visit the Notre-Dame Cathedral and climb up to the top for a stunning view of the city\n" +
"Explore the Latin Quarter and visit the Panthéon\n" +
"Go shopping at the famous Galeries Lafayette or Printemps department stores\n" +
"Have dinner at a local bistro\n" +
"\n" +
```

```
                "Day 6: Parisian Parks and Museums\n" +

                "Visit the Musée Rodin and explore its beautiful
gardens\n" +
                "Stroll through the Luxembourg Gardens and visit the
Luxembourg Palace\n" +
                "Visit the Centre Pompidou, a modern art museum in the
Marais neighborhood\n" +
                "Have dinner at a local restaurant in the Latin Quarter\n"
+
                "\n" +
                "Day 7: River Cruise and Farewell\n" +

                "Take a boat cruise along the Seine River to see the city
from a different perspective\n" +
                "Visit the Musée de l'Orangerie, which houses Monet's
famous water lilies paintings\n" +
                "Have a farewell dinner at a Michelin-starred restaurant"
        )
    }
}
```

RegisterActivity.kt

```
package com.example.travelapp

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.background
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.text.input.PasswordVisualTransformation
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import androidx.core.content.ContextCompat

class RegisterActivity : ComponentActivity() {
    private lateinit var databaseHelper: UserDatabaseHelper
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
```

```
        databaseHelper = UserDatabaseHelper(this)
        setContent {
            RegistrationScreen(this, databaseHelper)
        }
    }
}

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

    var username by remember { mutableStateOf("") }
    var password by remember { mutableStateOf("") }
    var email by remember { mutableStateOf("") }
    var error by remember { mutableStateOf("") }

    Column(
        modifier = Modifier.fillMaxSize().background(Color.White),
        horizontalAlignment = Alignment.CenterHorizontally,
        verticalArrangement = Arrangement.Center
    ) {

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

        Text(
```

```
    fontSize = 36.sp,
    fontWeight = FontWeight.ExtraBold,
    fontFamily = FontFamily.Cursive,
    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") },
            visualTransformation = PasswordVisualTransformation(),
            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)
}
```

SingaporeActivity.kt

```
package com.example.travelapp

import android.os.Bundle
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.rememberScrollState
import androidx.compose.foundation.verticalScroll
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.draw.scale
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.res.stringResource
import androidx.compose.ui.text.font.FontFamily
import androidx.compose.ui.tooling.preview.Preview
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
```

```
import com.example.travelapp.ui.theme.TravelAppTheme

class SingaporeActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            TravelAppTheme {
                // A surface container using the 'background' color from the
theme
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colors.background
                ) {
                    Greeting2()
                }
            }
        }
    }
}

@Composable
fun Greeting2() {

    Column(
        modifier = Modifier.background(color = Color.White)
            .padding(20.dp)
```

```
            .verticalScroll(rememberScrollState())
    ) {
        Text(
            fontSize = 40.sp,
            color = Color(android.graphics.Color.rgb(120, 40, 251)),
            fontFamily = FontFamily.Cursive,
            text = stringResource(id = R.string.place_3),
        )
        Image(
            painterResource(id = R.drawable.singapore),
contentDescription = "",
            modifier = Modifier
                .padding(16.dp)
                .fillMaxWidth()
                .height(200.dp)
                .scale(scaleX = 1.2F, scaleY = 1F)
        )
        Text(
            color = Color.Black,
            text = "Day 1:\n" +


                    "Morning: Visit Gardens by the Bay and marvel at the
Supertree Grove and the Flower Dome and Cloud Forest
conservatories.\n" +

                    "Afternoon: Explore the Marina Bay Sands complex,
which includes a casino, luxury shopping mall, and observation deck
with a stunning view of the city.\n" +
```

```
"\n" +
"Day 2:\n" +

"Morning: Explore the historic district of Chinatown, including the Buddha Tooth Relic Temple and Museum and the Sri Mariamman Temple.\n" +
"Afternoon: Visit the nearby Clarke Quay for lunch and to explore its waterfront restaurants, bars, and shops.\n" +
"\n" +
"Day 3:\n" +

"Morning: Take a tour of the UNESCO-listed Botanic Gardens, one of the world's most famous and significant tropical gardens.\n" +
"Afternoon: Head over to the National Museum of Singapore, which houses a vast collection of historical and cultural artifacts.\n" +

"\n" +
"Day 4:\n" +

"Morning: Visit the Singapore Zoo and admire the wildlife, including orangutans, tigers, and elephants.\n" +
"Afternoon: Head over to Sentosa Island and relax at one of its many beaches or try some of the many attractions such as Universal Studios Singapore or Adventure Cove Waterpark.\n" +
"\n" +
"Day 5:\n" +
```

```
                "Morning: Go on a nature walk at MacRitchie Reservoir, which offers hiking trails and stunning views of the city skyline.\n" +
                "Afternoon: Visit Little India, a vibrant and colorful neighborhood, and explore the shops, temples, and food stalls.\n" +
                "\n" +
                "Day 6:\n" +

                "Morning: Explore the trendy neighborhood of Tiong Bahru, known for its hip cafes and boutiques, as well as its Art Deco architecture.\n" +
                "Afternoon: Visit the National Gallery Singapore, which houses the largest public collection of modern art in Singapore and Southeast Asia.\n" +
                "\n" +
                "Day 7:\n" +

                "Morning: Take a day trip to the nearby island of Pulau Ubin, where you can rent a "
        )
    }
}
```

User.kt

```
package com.example.travelapp
```

```
import androidx.room.ColumnInfo
import androidx.room.Entity
import androidx.room.PrimaryKey

@Entity(tableName = "user_table")
data class User(
    @PrimaryKey(autoGenerate = true) val id: Int?,
    @ColumnInfo(name = "first_name") val firstName: String?,
    @ColumnInfo(name = "last_name") val lastName: String?,
    @ColumnInfo(name = "email") val email: String?,
    @ColumnInfo(name = "password") val password: String?,

    )
```

UserDao.kt

```
package com.example.travelapp

import androidx.room.*

@Dao
interface UserDao {

    @Query("SELECT * FROM user_table WHERE email = :email")
    suspend fun getUserByEmail(email: String): User?
```

```
    @Insert(onConflict = OnConflictStrategy.REPLACE)
    suspend fun insertUser(user: User)

    @Update
    suspend fun updateUser(user: User)

    @Delete
    suspend fun deleteUser(user: User)
}
```

UserDatabase.kt

```
package com.example.travelapp

import android.content.Context
import androidx.room.Database
import androidx.room.Room
import androidx.room.RoomDatabase

@Database(entities = [User::class], version = 1)
abstract class UserDatabase : RoomDatabase() {

    abstract fun userDao(): UserDao
```

```
    companion object {

        @Volatile
        private var instance: UserDatabase? = null

        fun getDatabase(context: Context): UserDatabase {
            return instance ?: synchronized(this) {
                val newInstance = Room.databaseBuilder(
                    context.applicationContext,
                    UserDatabase::class.java,
                    "user_database"
                ).build()
                instance = newInstance
                newInstance
            }
        }
    }
}
```

UserDatabaseHelper.kt

```
package com.example.travelapp

import android.annotation.SuppressLint
import android.content.ContentValues
```

```
import android.content.Context
import android.database.Cursor
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper

class UserDatabaseHelper(context: Context) :
    SQLiteOpenHelper(context, DATABASE_NAME, null,
DATABASE_VERSION) {

    companion object {
        private const val DATABASE_VERSION = 1
        private const val DATABASE_NAME = "UserDatabase.db"

        private const val TABLE_NAME = "user_table"
        private const val COLUMN_ID = "id"
        private const val COLUMN_FIRST_NAME = "first_name"
        private const val COLUMN_LAST_NAME = "last_name"
        private const val COLUMN_EMAIL = "email"
        private const val COLUMN_PASSWORD = "password"
    }

    override fun onCreate(db: SQLiteDatabase?) {
        val createTable = "CREATE TABLE $TABLE_NAME (" +
                "$COLUMN_ID INTEGER PRIMARY KEY
AUTOINCREMENT, " +
                "$COLUMN_FIRST_NAME TEXT, " +
```

```
            "$COLUMN_LAST_NAME TEXT, " +
            "$COLUMN_EMAIL TEXT, " +
            "$COLUMN_PASSWORD TEXT" +
            ")"

        db?.execSQL(createTable)
    }

    override fun onUpgrade(db: SQLiteDatabase?, oldVersion: Int,
newVersion: Int) {
        db?.execSQL("DROP TABLE IF EXISTS $TABLE_NAME")
        onCreate(db)
    }

    fun insertUser(user: User) {
        val db = writableDatabase
        val values = ContentValues()
        values.put(COLUMN_FIRST_NAME, user.firstName)
        values.put(COLUMN_LAST_NAME, user.lastName)
        values.put(COLUMN_EMAIL, user.email)
        values.put(COLUMN_PASSWORD, user.password)
        db.insert(TABLE_NAME, null, values)
        db.close()
    }

    @SuppressLint("Range")
```

```
    fun getUserByUsername(username: String): User? {
        val db = readableDatabase
        val cursor: Cursor = db.rawQuery("SELECT * FROM
$TABLE_NAME WHERE $COLUMN_FIRST_NAME = ?",
arrayOf(username))
        var user: User? = null
        if (cursor.moveToFirst()) {
            user = User(
                id = cursor.getInt(cursor.getColumnIndex(COLUMN_ID)),
                firstName =
cursor.getString(cursor.getColumnIndex(COLUMN_FIRST_NAME))
,
                lastName =
cursor.getString(cursor.getColumnIndex(COLUMN_LAST_NAME)),
                email =
cursor.getString(cursor.getColumnIndex(COLUMN_EMAIL)),
                password =
cursor.getString(cursor.getColumnIndex(COLUMN_PASSWORD)),
            )
        }
        cursor.close()
        db.close()
        return user
    }
    @SuppressLint("Range")
    fun getUserById(id: Int): User? {
        val db = readableDatabase
```

```
        val cursor: Cursor = db.rawQuery("SELECT * FROM
$TABLE_NAME WHERE $COLUMN_ID = ?",
arrayOf(id.toString()))
        var user: User? = null
        if (cursor.moveToFirst()) {
            user = User(
                id = cursor.getInt(cursor.getColumnIndex(COLUMN_ID)),
                firstName =
cursor.getString(cursor.getColumnIndex(COLUMN_FIRST_NAME))
,
                lastName =
cursor.getString(cursor.getColumnIndex(COLUMN_LAST_NAME)),
                email =
cursor.getString(cursor.getColumnIndex(COLUMN_EMAIL)),
                password =
cursor.getString(cursor.getColumnIndex(COLUMN_PASSWORD)),
            )
        }
        cursor.close()
        db.close()
        return user
    }

    @SuppressLint("Range")
    fun getAllUsers(): List<User> {
        val users = mutableListOf<User>()
        val db = readableDatabase
```

```
        val cursor: Cursor = db.rawQuery("SELECT * FROM
$TABLE_NAME", null)
        if (cursor.moveToFirst()) {
            do {
                val user = User(
                    id =
cursor.getInt(cursor.getColumnIndex(COLUMN_ID)),
                    firstName =
cursor.getString(cursor.getColumnIndex(COLUMN_FIRST_NAME))
,
                    lastName =
cursor.getString(cursor.getColumnIndex(COLUMN_LAST_NAME)),
                    email =
cursor.getString(cursor.getColumnIndex(COLUMN_EMAIL)),
                    password =
cursor.getString(cursor.getColumnIndex(COLUMN_PASSWORD)),
                )
                users.add(user)
            } while (cursor.moveToNext())
        }
        cursor.close()
        db.close()
        return users
    }


}
```

Color.kt

```
package com.example.travelapp.ui.theme

import androidx.compose.ui.graphics.Color

val Purple200 = Color(0xFFBB86FC)
val Purple500 = Color(0xFF6200EE)
val Purple700 = Color(0xFF3700B3)
val Teal200 = Color(0xFF03DAC5)
```

Shape.kt

```
package com.example.travelapp.ui.theme

import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.material.Shapes
import androidx.compose.ui.unit.dp

val Shapes = Shapes(
    small = RoundedCornerShape(4.dp),
    medium = RoundedCornerShape(4.dp),
    large = RoundedCornerShape(0.dp)
)
```

Theme.kt

```
package com.example.travelapp.ui.theme

import androidx.compose.foundation.isSystemInDarkTheme
import androidx.compose.material.MaterialTheme
import androidx.compose.material.darkColors
import androidx.compose.material.lightColors
import androidx.compose.runtime.Composable

private val DarkColorPalette = darkColors(
    primary = Purple200,
    primaryVariant = Purple700,
    secondary = Teal200
)

private val LightColorPalette = lightColors(
    primary = Purple500,
    primaryVariant = Purple700,
    secondary = Teal200

    /* Other default colors to override
    background = Color.White,
    surface = Color.White,
    onPrimary = Color.White,
```

```
    onSecondary = Color.Black,
    onBackground = Color.Black,
    onSurface = Color.Black,
    */
)

@Composable
fun TravelAppTheme(darkTheme: Boolean =
isSystemInDarkTheme(), content: @Composable () -> Unit) {
    val colors = if (darkTheme) {
        DarkColorPalette
    } else {
        LightColorPalette
    }

    MaterialTheme(
        colors = colors,
        typography = Typography,
        shapes = Shapes,
        content = content
    )
}
```

Type.kt

```
package com.example.travelapp.ui.theme

import androidx.compose.material.Typography
import androidx.compose.ui.text.TextStyle
import androidx.compose.ui.text.font.FontFamily
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.unit.sp

// Set of Material typography styles to start with
val Typography = Typography(
    body1 = TextStyle(
        fontFamily = FontFamily.Default,
        fontWeight = FontWeight.Normal,
        fontSize = 16.sp
    )
    /* Other default text styles to override
    button = TextStyle(
        fontFamily = FontFamily.Default,
        fontWeight = FontWeight.W500,
        fontSize = 14.sp
    ),
    caption = TextStyle(
        fontFamily = FontFamily.Default,
        fontWeight = FontWeight.Normal,
        fontSize = 12.sp
```

```
    )
    */
)
```