A Project Report On

An Android Application for keeping Up The Latest Headlines

BACHELOER OF COMPUTER SCIENCE

Devanger Arts college,

(Affiliated to Madurai Kamaraj University)

Aruppukottai-626101

Submitted

By

K.HEMARUBINI (Reg.No:C0S42001)

P.MUTHULAKSHMI (Reg.No:C0S42002)

S.VEERAKUMAR (Reg.No:C0S42022)

K.VIJAYAKUMAR (Reg.No:C0S42023)

Project Name:

Wanderlust: A Personalized Travel Planning And Tracking App

INTRODUCTION

1.1Overview

The My Travel Tracker app gives you the complete travel experience. The app has been designed by a solo traveling lady for travelers. A journal, a community, an inspiration feed, and much more to come. We are like an Instagram but completely focused on travel and extra travel related features..

1.2 Purpose

My Travel Tracker accompanies you on every step of your journey and inspires you for the ones to come. Share your adventures, save your favorite locations and expand your bucket list exploring different travel storie

A travel planning app is an application for booking travel reservations, tracking loyalty points, and browsing travel packages. A travel planning software package can be used by travel agencies, travel suppliers, and consumers

You can take photos, share them with friends and family and send them automatic updates about your travels in a few clicks. Plus, the app can syndicate your posts to Twitter and Facebook. You only need to share an update once or you can choose to keep it private and only send your messages to a private group of friends that you create on the platform. The app is available for free on android devices.

Problem Definition & Design Thinking

2.1 Empathy Map

Template

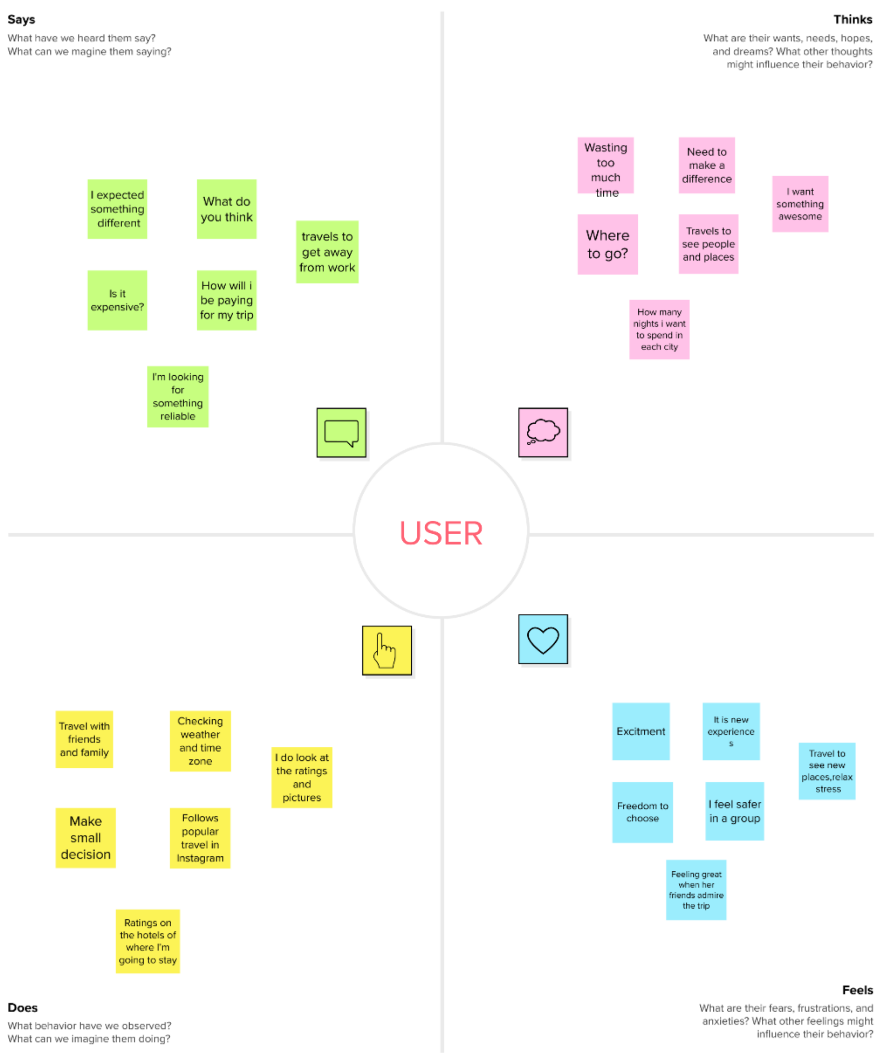
Empathy map

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

A Personalized Travel Planning And Tracking

Personalization in travel and experence sector is combined use of technology and traveler and tourist information to tailor electronic commerce interactions between service providers and each individual traveler or tourist



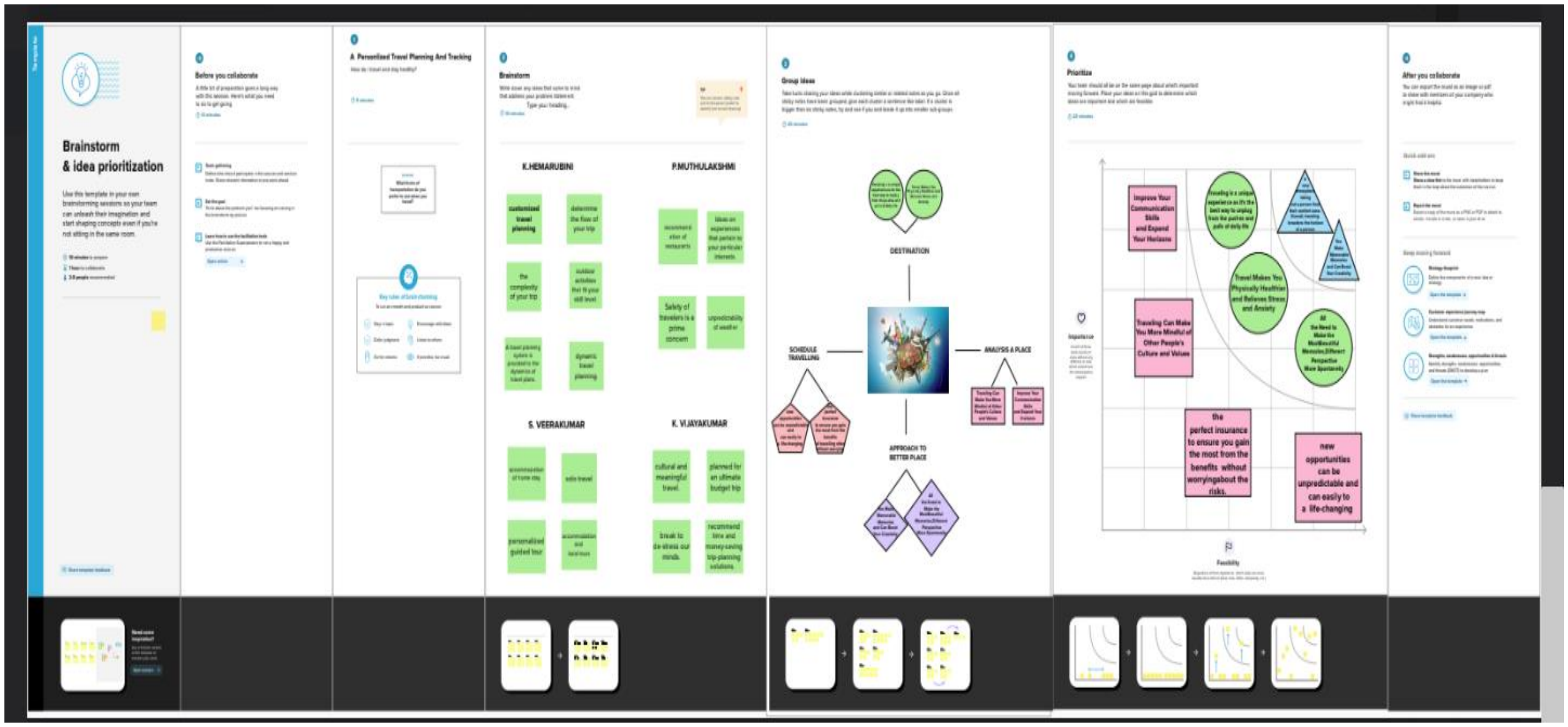
Says
What have we heard them say?
What can we magine them saying?
I expected something different
What do you think
travels to get away from work
Is it expensive?
How will i be paying for my trip
I'm looking for something reliable
Thinks
What are their wants, needs, hopes, and dreams? What other thoughts might influence their behavior?
Wasting too much time
Need to make a difference
I want something awesome
Where to go?
Travels to see people and places
How many nights i want to spend in each city
USER
Travel with friends and family
Checking weather and time zone
I do look at the ratings and pictures
Make small decision
Follows popular travel in Instagram
Ratings on the hotels of where I'm going to stay
Does
What behavior have we observed?
What can we imagine them doing?
Excitment
It is new experience s
Travel to see new places,relax stress
Freedom to choose
I feel safer in a group
Feeling great when her friends admire the trip
Feels
What are their fears, frustrations, and anxieties? What other feelings might influence their behavior?

Need some inspiration?

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

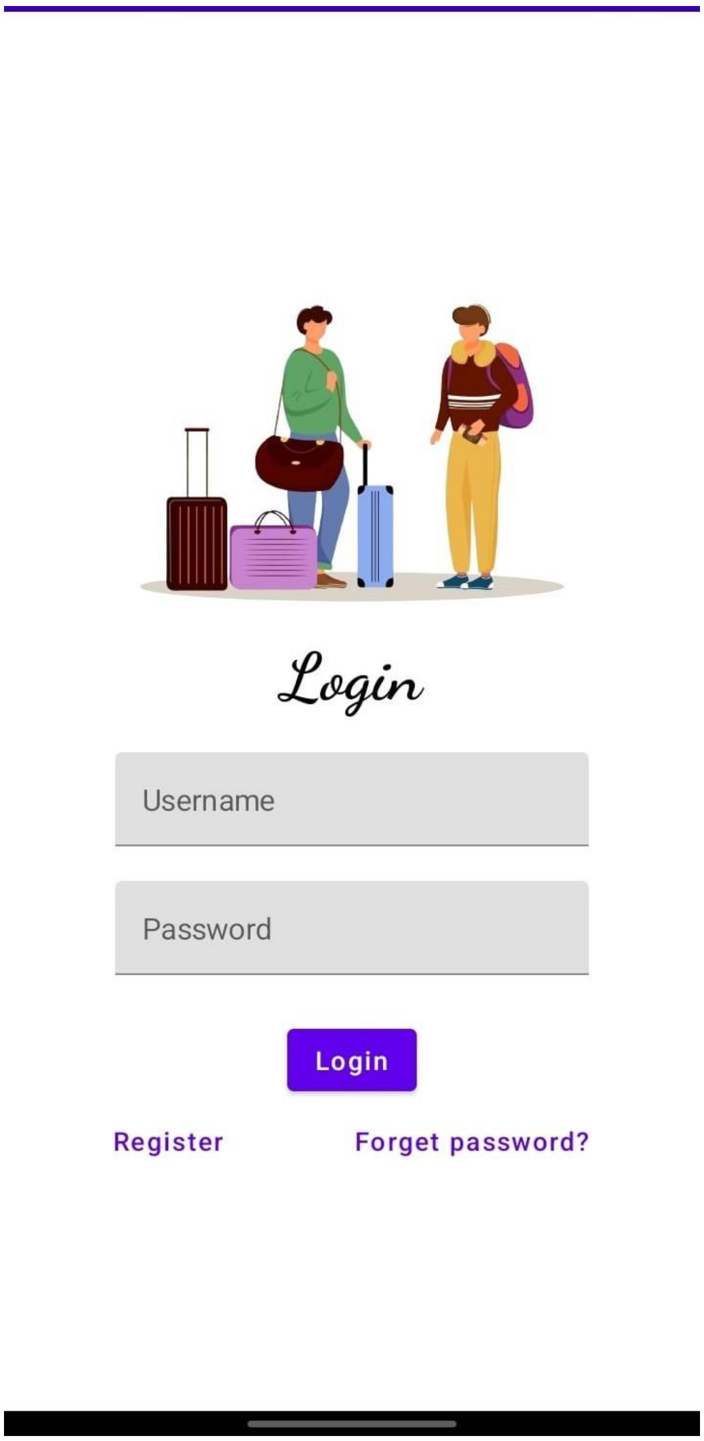
Open example →

2.2 Ideation & Brainstorming Map


Brainstorm & idea prioritization
Before you collaborate
A Personalized Travel Planning And Tracking
Brainstorm
K.HEMARUBINI
P.MUTHULAKSHMI
S. VEERAKUMAR
K. VIJAYAKUMAR
customized travel planning
determine the flow of your trip
the complexity of your trip
Safety of travelers is a prime concern
unpredictability of weather
dynamic travel planning
solo travel
personalized guided tour
cultural and meaningful travel
planned for an ultimate budget trip
break to de-stress our minds
Group ideas
DESTINATION
SCHEDULE TRAVELLING
ANALYSIS & PLACE
APPROACH TO BETTER PLACE
Prioritize
Improve Your Communication Skills and Expand Your Horizons
Travel Makes You Physically Healthier and Relieves Stress and Anxiety
Traveling Can Make You More Mindful of Other People's Culture and Values
the perfect insurance to ensure you gain the most from the benefits without worryingabout the risks.
new opportunities can be unpredictable and can easily to a life-changing
Importance
Feasibility
After you collaborate

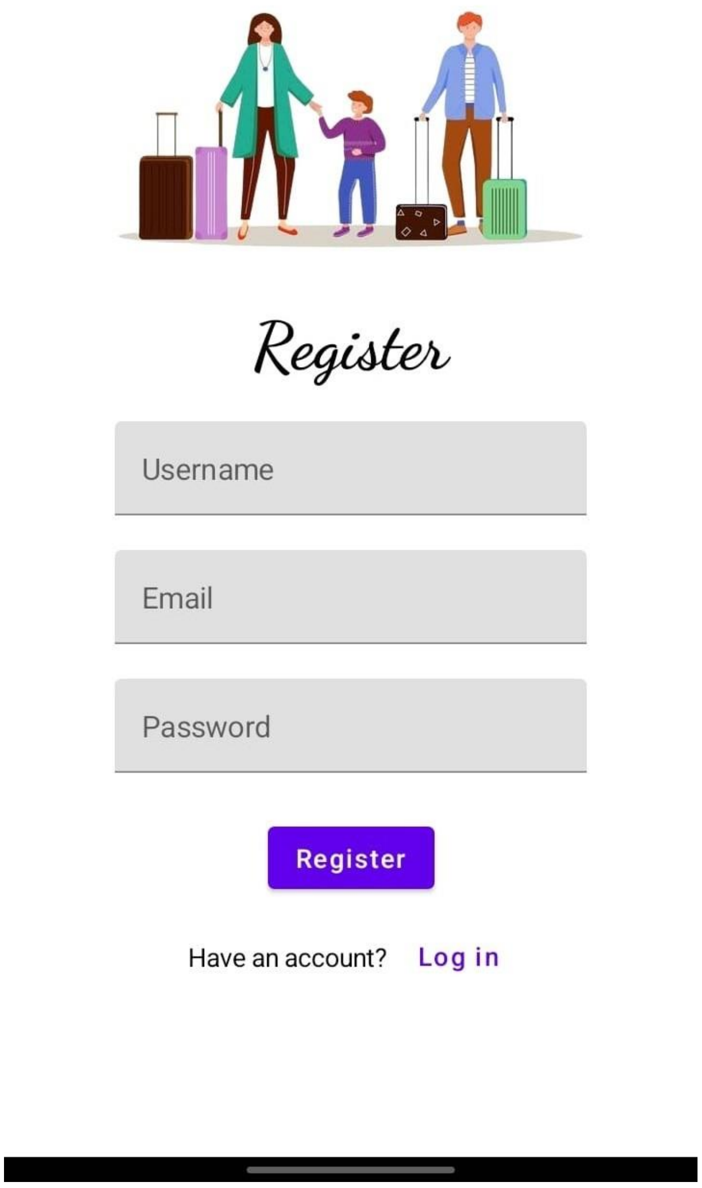
RESULT

Login Page :

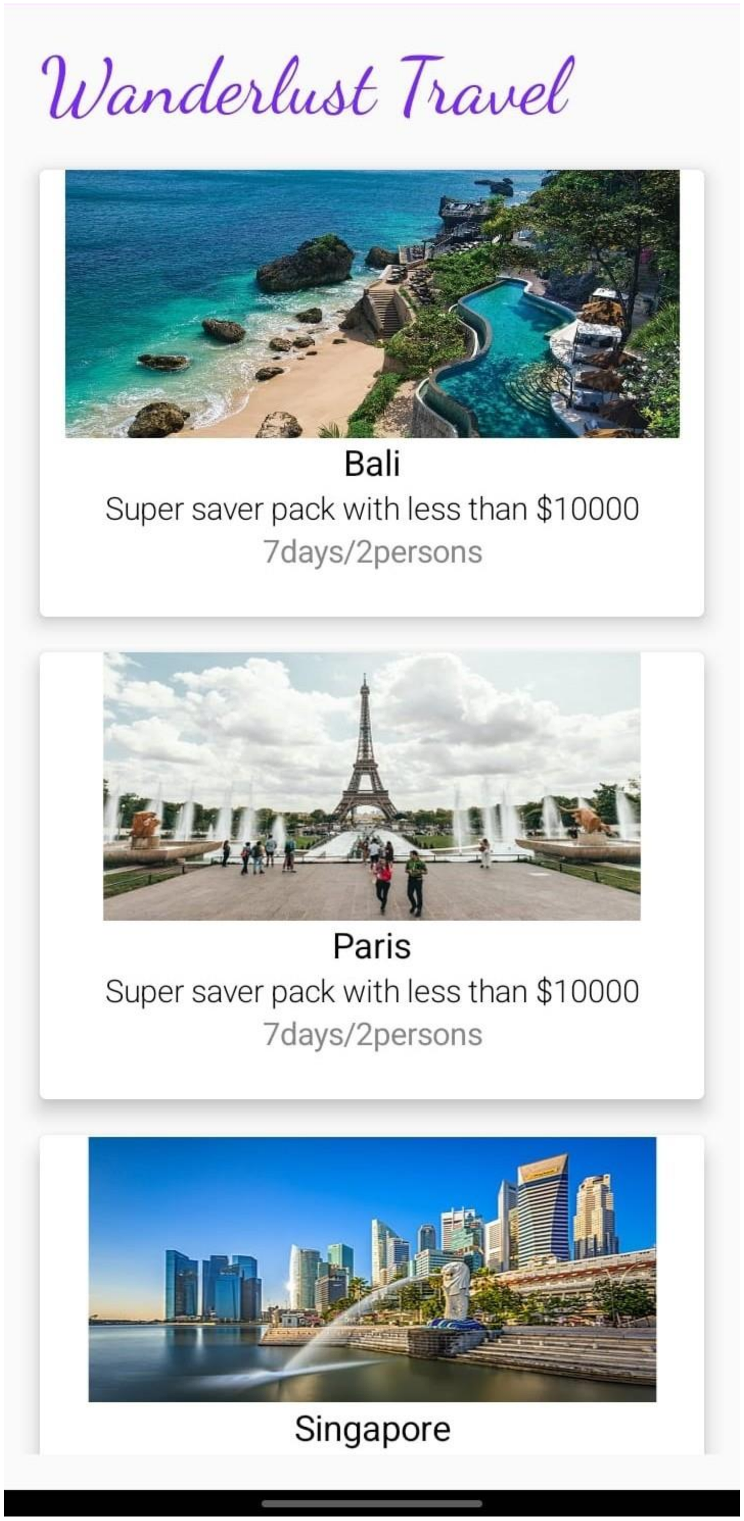


Login
Username
Password
Login
Register
Forget password?

RegisterPage :



MainPage :



Wanderlust Travel

Bali

Super saver pack with less than $10000

7days/2persons

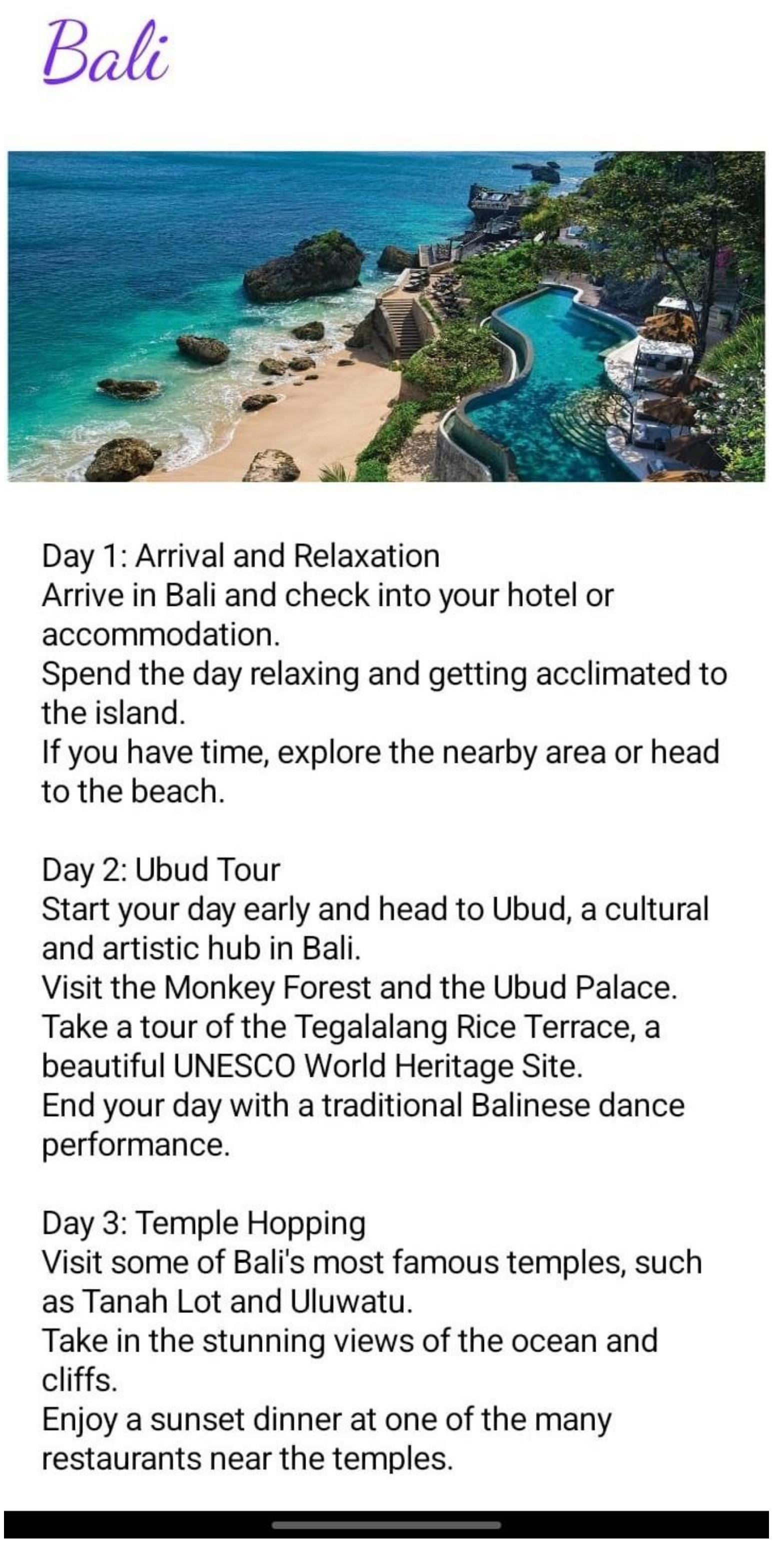
Paris

Super saver pack with less than $10000

7days/2persons

Singapore

Location Page:



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.
Take in the stunning views of the ocean and cliffs.
Enjoy a sunset dinner at one of the many restaurants near the temples.

ADVANTAGES

1. Communicate when it matters most

Quickly understanding the after-effect that the crisis might have on your people, organisation and assets is critical. Using an employee’s reallocation, your organisation can quickly locate and communicate with those impacted.

Relying on one sole channel for communication may not suffice in times of crisis. Due to overloaded cellular networks or internet outages, it is important to adopt various modes of communication from email, SMS, app notifications or text-to-speech.

2. Improves Safety and Security

Your Duty of Care is to the safety and security of your global workforce. Knowing where your employees are during a global security incident is vital. This will also help to facilitate support or potential evacuation should it be necessary.

Through an assistance app, your people can check in to proactively identify their location. Then tracking and monitoring allows your organisation to locate employees that may be impacted by a local incident. Your people impacted are then able to receive the required care and support.

3. Accelerate your Crisis Response

Should an incident occur, understanding the impact on your people and operations can be a time consuming and challenging task, especially when time is of the essence. With an effective crisis communications tool, a trigger will go off following a critical incident to identify which of your people are impacted. As a manager, you will be able to determine their safety with regular updates. As an organisation, you are prioritising the safety and security of your people, a critical part of an effective crisis management strategy.

DISADVANTAGES

1. Assuming always-on data connection.

It's not always on. Full stop.

Because it's mobile, users constantly lose coverage: metro/tube stations, inner buildings, car parkings, cathedrals (yes), rural areas.

And obviously, when it comes to traveling abroad, roaming forces data networks off for most users. Also let's not forget in-flight situations, where data is off and you'd still need to access data in your apps.

The fact is that as soon as coverage is lost, some apps stop working.

2. Assuming high speed network access.

Despite what you can read on theoretical data speeds, truth is mobile bandwidth is far from fast and predictable.

Tons of factors can influence the actual speed, such as number of people in the area, architectural structures of nearby buildings, weather conditions, and even the position of the hand on the device.

It's a good thing that our phones can transmit faster, but if everyone wants to transmit at the same time in a busy area, no one gets enough bandwidth and we must all keep retrying until everyone else is silent.

3. UX design too small and packed.

Some designers seem to forget that a finger is a finger. Every action button or dropdown list should be not only large enough that it can be hit with reasonable success, but also spaced away from other active items to avoid tapping the wrong one.

Selecting items in predictive search lists can be difficult if the results are too small and squeezed one next to the other.

APPLICATIONS

A myriad of tracking systems exists. Some are 'lag time' indicators, that is, the data is collected after an item has passed a point for example a bar codeor choke point or gate.[1] Others are 'real-time' or 'near real-time' like Global Positioning Systems (GPS) depending on how often the data is refreshed. There are bar-code systems which require items to be scanned an auto-id). For the most part, the tracking worlds are composed of discrete hardware and software systems for different applications. That is, bar-code systems are separate from Electronic Product Cod (EPC) systems, GPS systems are separate from active real time locating systems or RTLS for example, a passive RFID system would be used in a warehouse to scan the boxes as they are loaded on a truck - then the truck itself is tracked on a different system using GPS with its own features and software.[2 The major technology “silos” in the supply chain are:

Distribution /warehousing /manufacturing

Indoors assets are tracked repetitively reading e.g. a barcode,[3] any passive and active and feeding read data into Work in Progress models (WIP) or Warehouse Management Systems (WMS) or ERP software. The readers required per choke point are meshed auto-ID or hand-held ID applications.

However tracking could also be capable of providing monitoring data without binding to a fixed location by using a cooperative tracking capability, e.g. an RTLS.

Yard management

Outdoors mobile assets of high value are tracked by choke point,[4] 802.11, Received Signal Strength Indication (RSSI), Time Delay on Arrival (TDOA), active RFID or GPS Yard Management; feeding into either third party yard management software from the provider or to an existing system. Yard Management Systems (YMS) couple location data collected by RFID and GPS systems to help supply chain managers to optimize utilization of yard assets such as trailers and dock doors. YMS systems can use either active or passive RFID tags.

Fleet management

Fleet management is applied as a tracking application using GPS and composing tracks from subsequent vehicle's positions. Each vehicle to be tracked is equipped with a GPS receiver and relays the obtained coordinates via cellularor satellite networks to a home station.[5] Fleet management is required by:

- Large fleet operators, (vehicle/railcars/trucking/shipping)
- Forwarding operators (containers, machines, heavy cargo, valuable shippings)
- Operators who have high equipment and/or cargo/product costs
- Operators who have a dynamic workload

Person tracking

See also: Real-time locating system Targeted surveillanceCell phone surveillance and Digital contact tracing

Person tracking relies on unique identifiers that are temporarily (RFID tags) or permanently assigned to persons like personal identifiers (including biometric identifiers), or national identification numbers and a way to sample their positions, either on short temporal scales as through GPS or for public administration to keep track of a state's citizens or temporary residents. The purposes for doing so are numerous, for example from welfare and public security to mass surveillance

CONCLUSION

It's no wonder that, the app development process is exhausting, and overwhelming also. There have been lots of steps and processes to follow, and many decision making mechanism are involved as well.

But, we have tried our best to cover multiple scenarios and possibilities you might encounter. There are still challenges that could be out of the box, but this guidebook will hopefully help you create the basic framework for the process as well as the application.

FUTURE SCOPE

Geo-Tracking Services

A vital goal for hitting the gold mine in travel app development comes from the successful integration of GPS based location services. After dropping at a certain destination the first thing that a tourist wants to check out is a hotel, cafe or a nearby local attraction. Your travel app should get a GPS lock, track your location and quickly fetch results. Mobile apps such as RunGo does a fine job of searching nearby locations for all kinds of services a tourist can avail while on a trip.

Weather or Climate Forecasting

A lot of tourists book their flights in haste not thinking about the ramifications that a climate change can bring on an unplanned journey. To prevent such calamities travel mobile app development companies should always integrate a real-time climate forecast. Just to alert the traveler of upcoming weather predictions, when they are creating their travel itinerary. To get an idea of what I am talking about, check out apps like AccuWeather, that displays weather reports for a vast number of locations all over the world. It let users check cloud formation patterns, wind speed, humidity and several other important factors before planning trips.

Travel Itinerary Generator

This is the most important feature demanded by travel enthusiasts or tourists. What makes it so special?

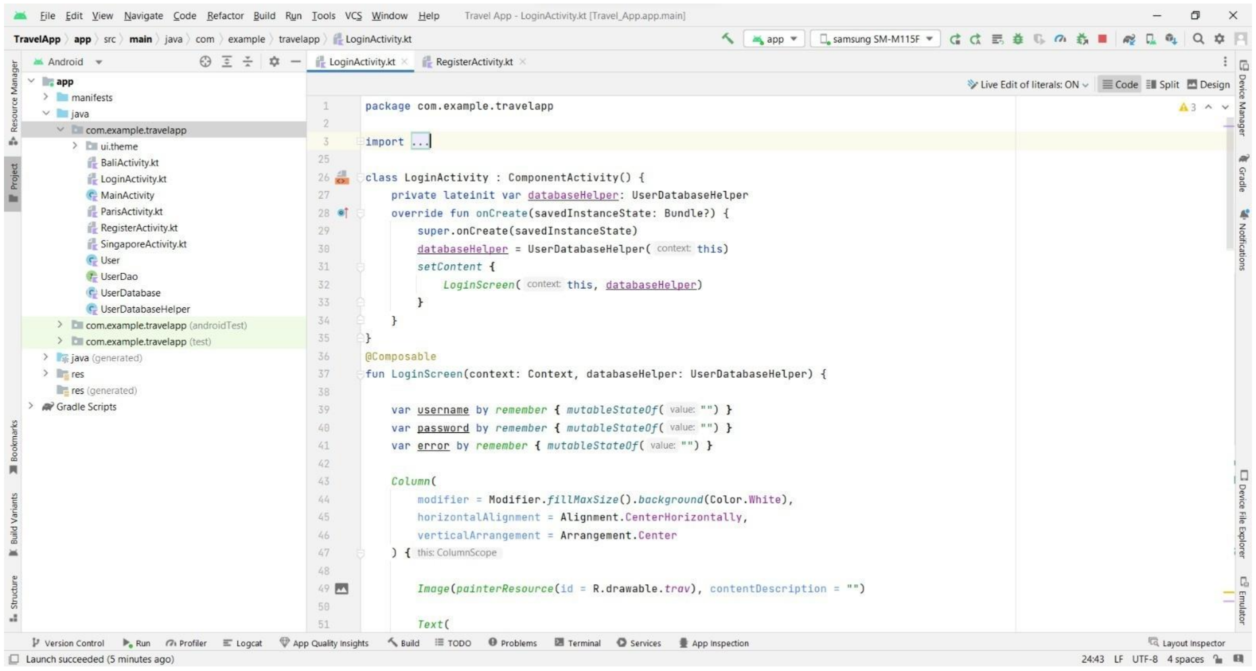
An efficient itinerary generator only asks the user to mention the locations that they want to visit and automatically creates a travel plan from it.

Take a look at services like TripHobo, that lets users add existing tourist attractions or even add custom landmarks to generate their trip plan accordingly. Implementing such complex algorithms is not easy. But if you want to make your travel app a successful, then ask your travel app developers to work on it right on!

APPENDIX

SOURCE CODE

LoginActivity



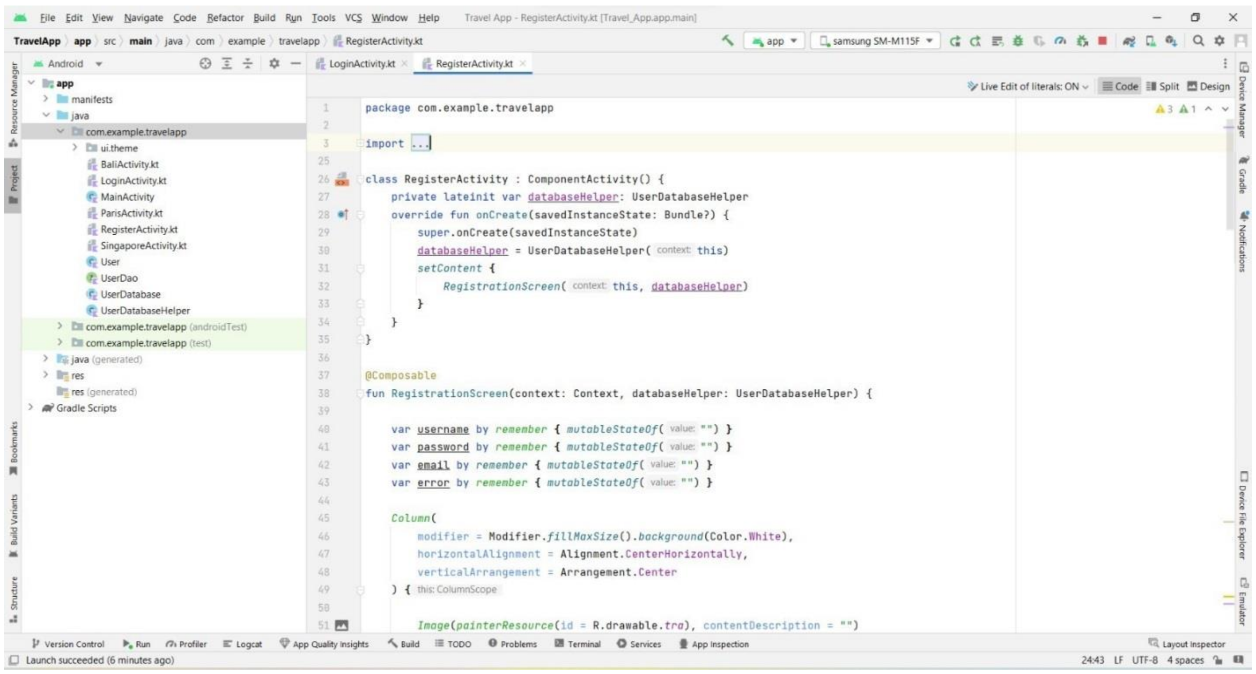
```
package com.example.travelapp

import ...

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

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

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

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

        Text(
```

Launch succeeded (5 minutes ago)

```
Packagecom.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)
}
```

RegisterActivity:



```
package com.example.travelapp

import ...

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 = "")
```

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

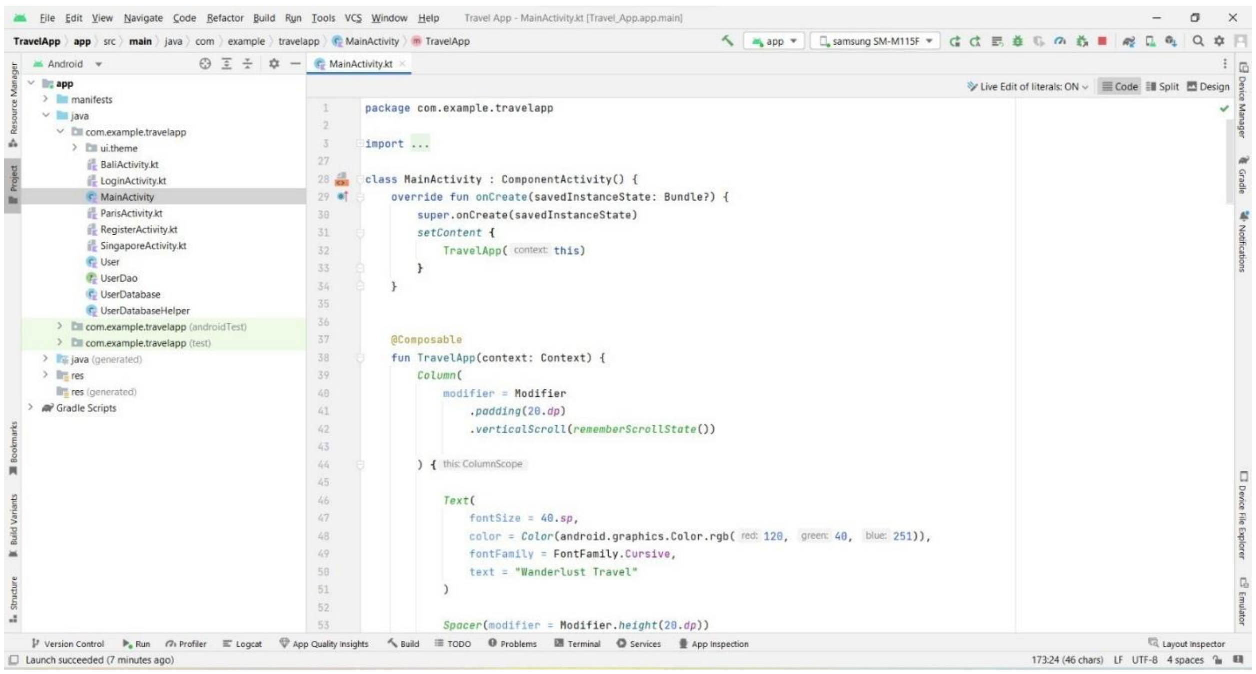
```
import android.content.Context
import android.content.Intent
import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.Image
import androidx.compose.foundation.background
import androidx.compose.foundation.layout.*
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)
    }
```

MainActivity:



```kotlin
package com.example.travelapp

import ...

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


    @Composable
    fun TravelApp(context: Context) {
        Column(
            modifier = Modifier
                .padding(20.dp)
                .verticalScroll(rememberScrollState())

        ) { this: ColumnScope

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

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

Launch succeeded (7 minutes ago)

```
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))
```

```
            }
        }
    }
```