Wanderlust: A Personalized Travel Planning And Tracking App

Submitted by

DATLA ADITYA VARMA	-20981A4614
CH RAHUL	-20981A4917
MUDDANA CHAITANYA ABHIRAM	-20981A4933
MARISERLA SANDEEP KUMAR	-21985A4904

Submitted to

INTRODUCTION

1.1 Overview

The "Travel Website" has been developed to override the problems prevailing in the practicing manual system. This software is supported to eliminate and in some cases reduce the hardships faced by this existing system. Moreover this system is designed for the particular need of the company to carry out operations in a smooth and effective manner.

The application is reduced as much as possible to avoid errors while entering the data. It also provides error message while entering invalid data. No formal knowledge is needed for the user to use this system. Thus by this all it proves it is user-friendly. Travel Website, as described above, can lead to error free, secure, reliable and fast management system. It can assist the user to concentrate on their other activities rather to concentrate on the record keeping. Thus it will help organization in better utilization of resources.

Every organization, whether big or small, has challenges to overcome and managing the information of Tour, Travel, Hotel, Customer, Facility. Every Travel Website has different Travel needs, therefore we design exclusive employee management systems that are adapted to your managerial requirements. This is designed to assist in strategic planning, and will help you ensure that your organization is equipped with the right level of information and details for your future goals. Also, for those busy executive who are always on the go, our systems come with remote access features, which will allow you to manage your workforce anytime, at all times. These systems will ultimately allow you to better manage resources.

1.2 Scope of the project

It may help collecting perfect management in details. In a very short time, the collection will be obvious, simple and sensible. It will help a person to know the management of passed year perfectly and vividly. It also helps in current all works relative to Travel Website. It will be also reduced the cost of collecting the management & collection procedure will go on smoothly.

Our project aims at Business process automation, i.e. we have tried to computerize various processes of Travel Website.

In computer system the person has to fill the various forms & number of copies of the forms can be easily generated at a time.

To assist the staff in capturing the effort spent on their respective working areas.

To utilize resources in an efficient manner by increasing their productivity through automation.

1.3 Reports of Travel Website:

It generates the report on Travel, Tour, Customer

- Provide filter reports on Agent, Hotel, Facility

You can easily export PDF for the Travel, Customer, Hotel

Application also provides excel export for Tour, Agent, Facility

- You can also export the report into csv format for Travel, Tour, Facility

1.3 Modules of Travel Website:

Travel Management Module: Used for managing the Travel details.

Facility Module: Used for managing the details of Facility

Customer Module: Used for managing the details of Customer

Tour Management Module: Used for managing the information and details of the Tour.

Agent Module: Used for managing the Agent details

Hotel Module: Used for managing the Hotel information

Login Module: Used for managing the login details

1.4 Input Data and Validation of Project on Travel Website

All the fields such as Travel, Agent, Facility are validated and does not take invalid values

Each form for Travel, Tour, Customer can not accept blank value fields

Avoiding errors in data

Controlling amount of input

Integration of all the modules/forms in the system.

Preparation of the test cases.

Preparation of the possible test data with all the validation checks.

Actual testing done manually.

Recording of all the reproduced errors.

Modifications done for the errors found during testing.

Prepared the test result scripts after rectification of the errors.

Functionality of the entire module/forms.

Validations for user input.

1.4 Software Requirement Specification

The Software Requirements Specification is produced at the culmination of the analysis task.

The function and performance allocated to software as part of system engineering are refined by

establishing a complete information description, a detailed functional and behavioral description, an

indication of performance requirements and design constraints, appropriate validation criteria,

another data pertinent to requirements.

The proposed system has the following requirements:

System needs store information about new entry of Travel

System needs to help the internal staff to keep information of Tour and find them as per various queries.

System need to maintain quantity record.

System need to keep the record of Agent

System need to update and delete the record.

System also needs a search area.

Advantages:

1. Organization: A travel plan helps you stay organized, ensuring you don't miss out on important details and destinations during your trip.
2. Cost-effectiveness: A well-thought-out travel plan can help you budget your expenses better and avoid overspending.
3. Time efficiency: With a plan in place, you can make the most of your time by knowing where to go and what to do without wasting time on indecision.
4. Reduced stress: Having a travel plan can reduce stress as you know what to expect and have contingencies for unforeseen events.
5. Better experience: A well-crafted travel plan allows you to cover all the must-visit spots and activities, enhancing your overall travel experience.

Disadvantages:

1. Rigidity: Following a strict travel plan may limit your spontaneity and flexibility to explore unexpected opportunities.
2. Unrealistic expectations: Overplanning can lead to high expectations, and if things don't go as planned, it may lead to disappointment.
3. Missed opportunities: Sometimes, sticking to a plan might cause you to miss out on hidden gems and off-the-beaten-path experiences.
4. Overcrowded destinations: Popular tourist spots mentioned in your plan may be crowded, reducing the charm of your visit.
5. Unexpected changes: Travel plans can be affected by unforeseen circumstances like

weather disruptions or transportation delays, causing adjustments to the itinerary.

RESULT

Travel plan websites are valuable tools that can help travelers create and manage their itineraries effectively. Here are some common uses of travel plan websites:

1. Itinerary Creation: Travel plan websites allow users to create detailed itineraries for their trips. They can input the destinations they want to visit, activities they want to do, and the duration of their stay.

2. Destination Research: These websites often provide comprehensive information about various destinations, including popular attractions, local customs, best times to visit, and travel tips.

3. Budgeting and Cost Estimation: Travel plan websites may offer budgeting features that help travelers estimate their expenses, including accommodation, transportation, food, and activities, allowing them to plan accordingly.

4. Scheduling and Time Management: Users can organize their travel schedules efficiently, ensuring they make the most of their time at each location.

5. Collaboration and Sharing: Some travel plan websites enable users to collaborate with others, making it easy for groups of travelers to plan their trips together. They can also share their itineraries with friends and family.

6. Real-Time Updates: Some websites provide real-time updates on weather conditions, flight schedules, and local events, helping travelers adapt their plans if needed.

7. Booking Services: Many travel plan websites integrate booking options for accommodations, flights, and activities, streamlining the booking process for users.

8. Offline Access: Some platforms offer offline access to itineraries, ensuring travelers can access their plans even without an internet connection.

9. Customization: Travelers can tailor their plans based on their preferences, interests, and limitations, ensuring a personalized and enjoyable experience.

10. Reviews and Recommendations: Travel plan websites often have user-generated content, including reviews and recommendations, providing valuable insights for planning a trip.

Using a travel plan website can enhance the travel experience by providing a centralized platform for all trip-related information and simplifying the planning process, making it easier to explore new destinations with confidence.

APPLICATIONS

Travel plan websites have various applications that cater to different aspects of travel planning and exploration. Some of the key applications include:

1. Vacation Planning: Travel plan websites are primarily used for planning vacations and trips. Users can create detailed itineraries, select destinations, book accommodations, and schedule activities.

2. Solo Travel: Individuals traveling alone can use travel plan websites to create personalized itineraries, find solo-friendly destinations, and connect with other solo travelers.

3. Group Travel: Travel plan websites facilitate group coordination, allowing multiple travelers to collaborate on the itinerary, share ideas, and ensure everyone's preferences are considered.

4. Business Travel: For business travelers, these websites can help with organizing meetings, finding appropriate accommodations, and planning leisure activities during downtime.

5. Adventure Travel: Travel plan websites often cater to adventure seekers, offering information on thrilling destinations, outdoor activities, and adventure-focused itineraries.

6. Family Vacations: Families can use these websites to plan family-friendly activities, identify kid-friendly attractions, and manage the logistics of traveling with children.

7. Honeymoon Planning: Travel plan websites can assist in creating romantic getaways, suggesting honeymoon destinations, and arranging special experiences for newlyweds.

8. Backpacking Trips: Backpackers can benefit from travel plan websites by discovering budget-friendly accommodations, transportation options, and off-the-beaten-path destinations.

9. Road Trips: These websites can assist in mapping out road trip routes, identifying scenic drives, and locating rest stops and attractions along the way.

10. Cultural Tours: Travel plan websites often feature cultural tours and heritage destinations, providing insights into local traditions, history, and cultural experiences.

11. Study Abroad Planning: Students planning to study abroad can use travel plan websites to prepare their travel arrangements, learn about their destination, and connect with other students.

12. Last-Minute Getaways: In situations where travelers need to plan a quick getaway, these websites can offer ready-made itineraries and suggestions for spontaneous trips.

Overall, travel plan websites are versatile tools that cater to various travel preferences, making it easier for individuals and groups to create well-organized and enjoyable travel

experiences.

CONCLUSION

In conclusion, travel plan websites play a crucial role in enhancing the travel experience for individuals and groups alike. These websites offer a wide range of applications, from vacation planning and solo travel to family vacations and adventure trips. By providing valuable destination information, itinerary creation tools, budgeting features, and real-time updates, they simplify the planning process and enable travelers to make informed decisions.

Travel plan websites offer the convenience of centralized travel management, allowing users to access all trip-related information in one place. They promote better organization, time efficiency, and cost-effectiveness, helping travelers create well-structured itineraries that maximize their exploration opportunities while reducing stress and uncertainty.

Additionally, these platforms cater to various travel styles, from backpacking and road trips to cultural tours and business travel, ensuring they can meet the diverse needs and preferences of travelers worldwide. Whether users seek spontaneous getaways or meticulously planned adventures, travel plan websites offer valuable resources and tools to make their journeys more enjoyable and memorable.

As technology continues to advance, travel plan websites are likely to evolve further, incorporating new features and improving user experiences. Overall, these websites have become indispensable tools for modern travelers, empowering them to embark on their journeys with confidence and discovertheworldwithease.

APPENDIX

SourceCode:

```
<?xmlversion="1.0"encoding="utf-8"?>
<manifestxmlns: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>
                <actionandroid:name="android.intent.action.MAIN" />
```

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

</manifest>
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."
        )


    }
}
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)
}
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)
}
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)
}
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"
        )
    }
}
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)
}
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
colorfulneighborhood, 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 "
        )
    }
}
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") valfirstName: String?,
    @ColumnInfo(name = "last_name") vallastName: String?,
    @ColumnInfo(name = "email") val email: String?,
    @ColumnInfo(name = "password") val password: String?,
)
package com.example.travelapp

import androidx.room.*

@Dao
interface UserDao {

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

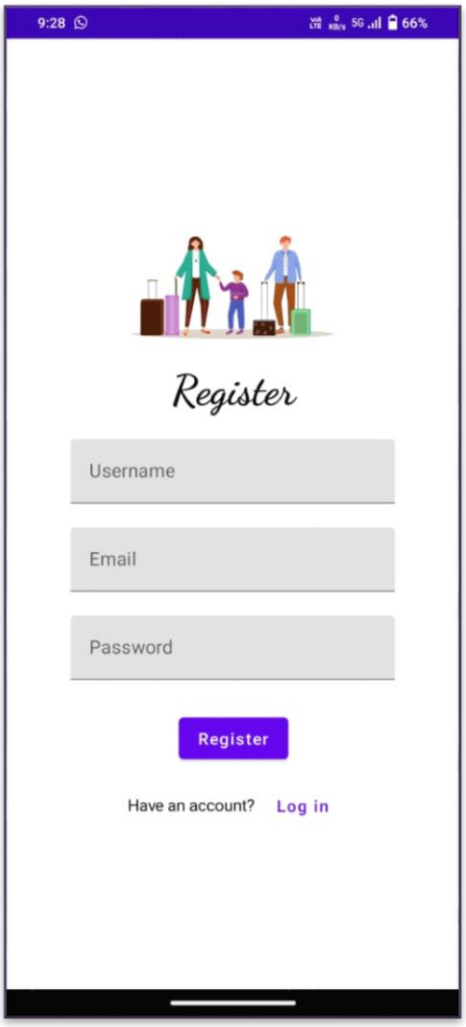
```
    @Insert(onConflict = OnConflictStrategy.REPLACE)
    suspend fun insertUser(user: User)

    @Update
    suspend fun updateUser(user: User)

    @Delete
    suspend fun deleteUser(user: User)
}
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)
}
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) {
valnewInstance = Room.databaseBuilder(
context.applicationContext,
UserDatabase::class.java,
                    "user_database"
).build()
                instance = newInstance
```

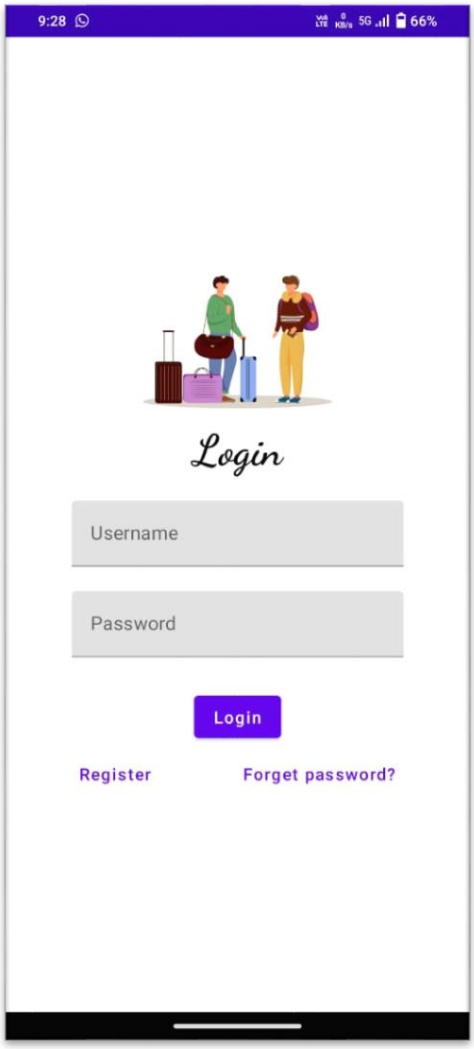
```
newInstance
            }
        }
    }
}
package com.example.travelapp

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 constval DATABASE_VERSION = 1
        private constval DATABASE_NAME = "UserDatabase.db"

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

    override fun onCreate(db: SQLiteDatabase?) {
valcreateTable = "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) {
valdb = 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)
```

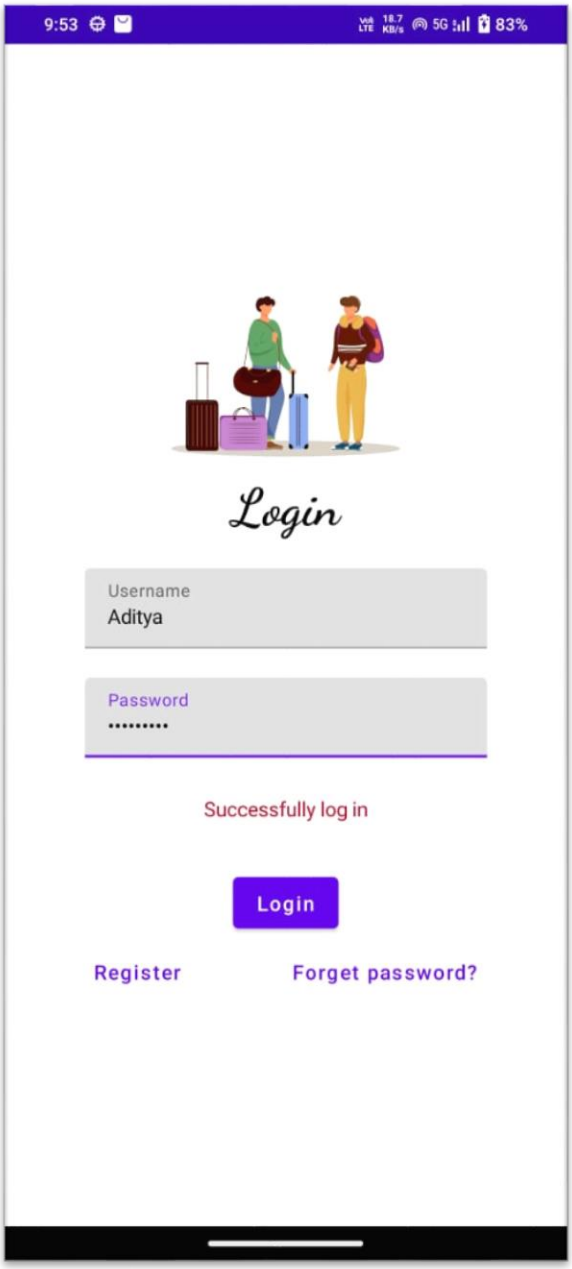
```
db.insert(TABLE_NAME, null, values)
db.close()
    }

    @SuppressLint("Range")
    fun getUserByUsername(username: String): User? {
valdb = 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? {
valdb = 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>()
valdb = 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)),
```

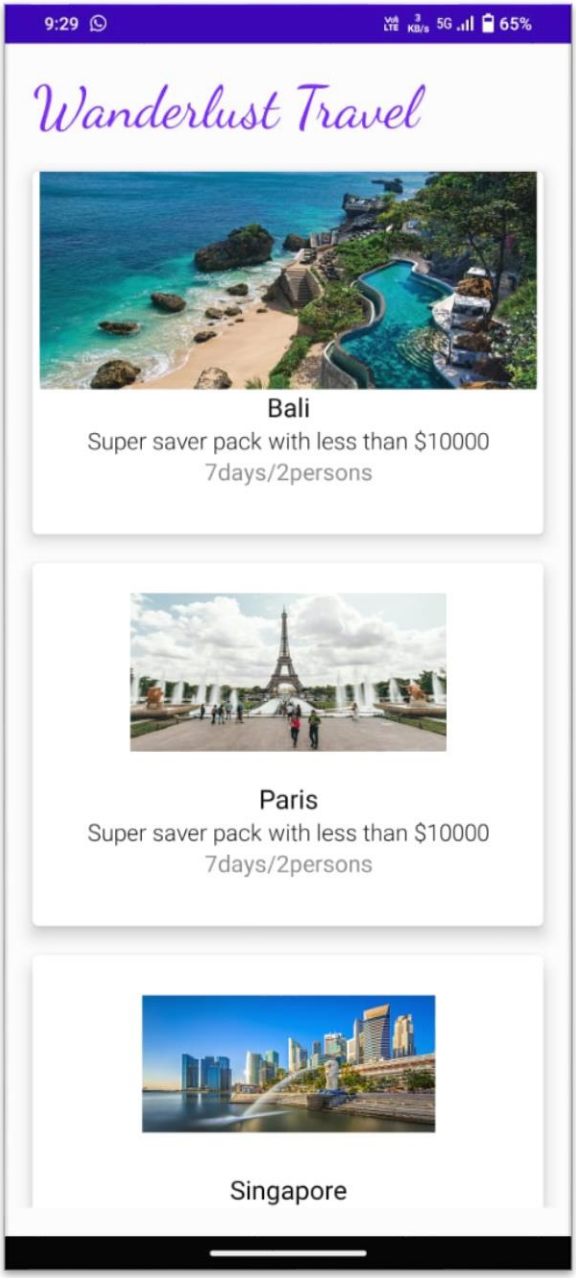
```
lastName = cursor.getString(cursor.getColumnIndex(COLUMN_LAST_NAME)),
                    email = cursor.getString(cursor.getColumnIndex(COLUMN_EMAIL)),
                    password =
cursor.getString(cursor.getColumnIndex(COLUMN_PASSWORD)),
                )
users.add(user)
            } while (cursor.moveToNext())
        }
cursor.close()
db.close()
        return users
    }

}
```

OUT PUT :


9:28
66%
Register
Username
Email
Password
Register
Have an account?
Log in


9:28
66%
Login
Username
Password
Login
Register
Forget password?


9:53
83%
Login
Username
Aditya
Password
Successfully log in
Login
Register
Forget password?


9:29
65%
Wanderlust Travel
Bali
Super saver pack with less than $10000
7days/2persons
Paris
Super saver pack with less than $10000
7days/2persons
Singapore


9:29
65%
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 Place du Tertre
Explore the historic neighborhood of Le Marais
Try some delicious French pastries at a local bakery
Have dinner at a brasserie to taste some classic French cuisine