Take A Trip

Table of contents

Chapter 1 - Introduction

1.1 Need For Product

Pakistan is a country that is rich in natural beauty and historical landmarks, which attracts numerous tourists from various nations. However, traveling to a foreign country can be overwhelming, and tourists often face several challenges in terms of finding the right information and resources to make their trip comfortable and enjoyable. To address this issue, we have developed an mobile application that provides a solution to some of the traveler's needs.

Our application's significant module is designed to provide essential information about the routes, places of interest, and necessary facilities for tourists, particularly for foreigners coming to Pakistan. The application's primary aim is to help travelers explore the natural beauty and historical landmarks of Pakistan without facing any difficulties.

1.2 Benefits to Users

Our mobile application provides numerous benefits to users, particularly tourists (foreigners) visiting Pakistan. The following section highlights some of the key advantages that users can expect to experience when using our application:

- **Convenience:**
  Our mobile application provides users with a one-stop-shop for all their travel needs, including packing assistance, hotel recommendations, trip planning, and navigation services, all from the convenience of their mobile device. This feature eliminates the need for users to consult multiple sources for information, making their travel experience more convenient and hassle-free.

- **Time-Saving:**
  With our trip planner feature, users can save time by planning their trips efficiently on their mobile device, whether they're waiting for their flight or exploring a new city. They can explore the places of interest, routes, and travel recommendations without spending hours researching on their own.

- **Cost-Effective:**
  Our mobile application's travel expense calculator and currency conversion tools provide users with a cost-effective way to manage their finances during their trip, right from their mobile device. They can keep track of their expenses in local

currency and convert it to their home currency, helping them to budget their expenses effectively.

- **Safety and Security:**
  Our mobile application's location-based emergency services feature provides users with a safety net in case of any emergency, accessible directly from their mobile device. They can access emergency services and contact the relevant authorities if needed, ensuring their safety and security during their travels.

- **Personalization:**
  Our mobile application allows users to customize their travel plans according to their preferences, interests, and budget, all through their mobile device. With our share trip feature, they can also share their trip details with their family and friends, keeping them updated on their travels and making it easier to plan trips with others.

1.3 Gap Analysis with Existing Solutions

The travel industry is witnessing a rapid shift towards mobile applications that provide travelers with convenient and personalized travel solutions. In this context, we have analyzed some of the existing travel applications, including Tripadvisor, skyScanner, PackPoint, and Visit a City, to identify gaps in their features and services compared to our proposed application.

- **TripAdvisor** is a popular travel application that provides travelers with user-generated reviews, recommendations, and bookings for hotels, restaurants, and attractions. However, it lacks several features that our proposed application offers, such as packing assistance, trip planning, and navigation services. Our application provides users with a one-stop-shop for all their travel needs, including essential information about the best routes, places of sight, and other travel recommendations, making it a more comprehensive and convenient solution for travelers.

- **SkyScanner** is another popular application that provides flight booking services, but it lacks several features such as packing assistance, trip planning, and personalized recommendations, which are essential for travelers. Our application offers all these features and services, making it a more comprehensive and convenient travel solution for users.

- **PackPoint** is an application that helps users create packing lists based on their destination, travel dates, and planned activities. However, it lacks several features such as hotel recommendations, trip planning, and navigation services, which are crucial for travelers.

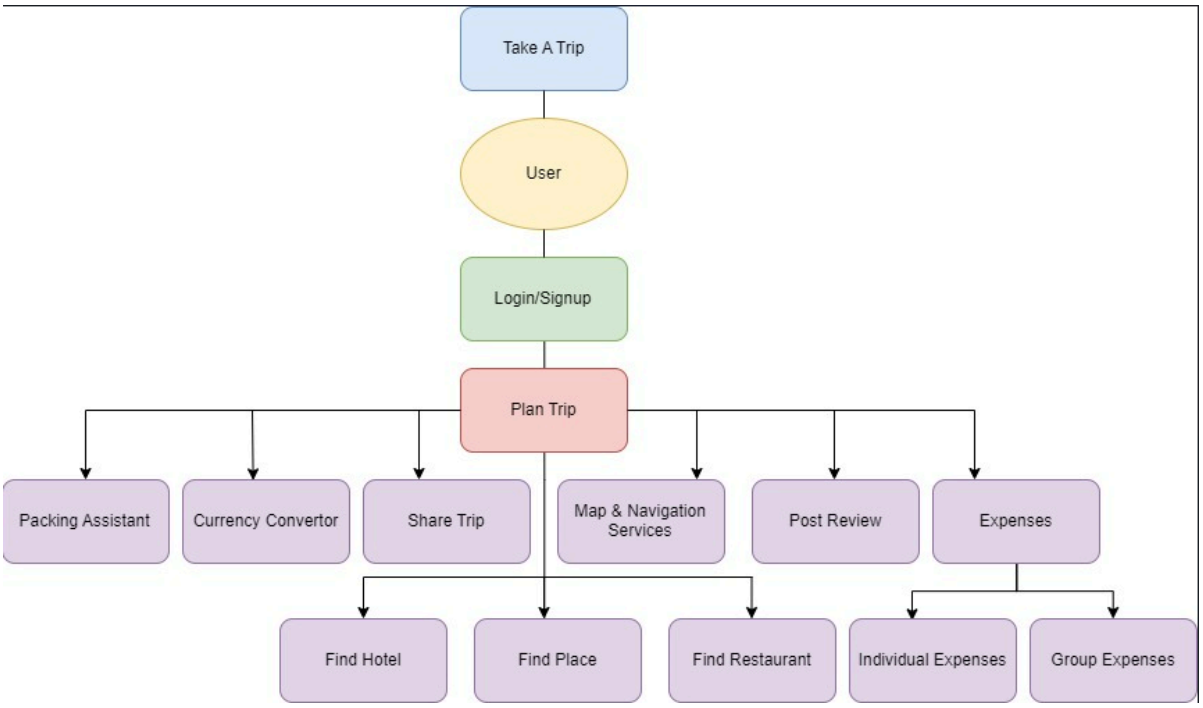
- **Visit a City** is an application that provides travelers with suggested itineraries for exploring different cities around the world. However, it lacks several features such as packing assistance, hotel recommendations, and travel expense calculator, which are crucial for travelers. Our application provides users with a more comprehensive and convenient travel solution by offering all these features and services in one place.

In conclusion, while existing travel applications offer some useful features and services, they lack several critical features that our proposed application offers, such as packing assistance, hotel recommendations, trip planning, navigation services, travel expense calculator, and location-based emergency services. By addressing these gaps, our application aims to provide travelers with a more comprehensive and convenient travel solution.

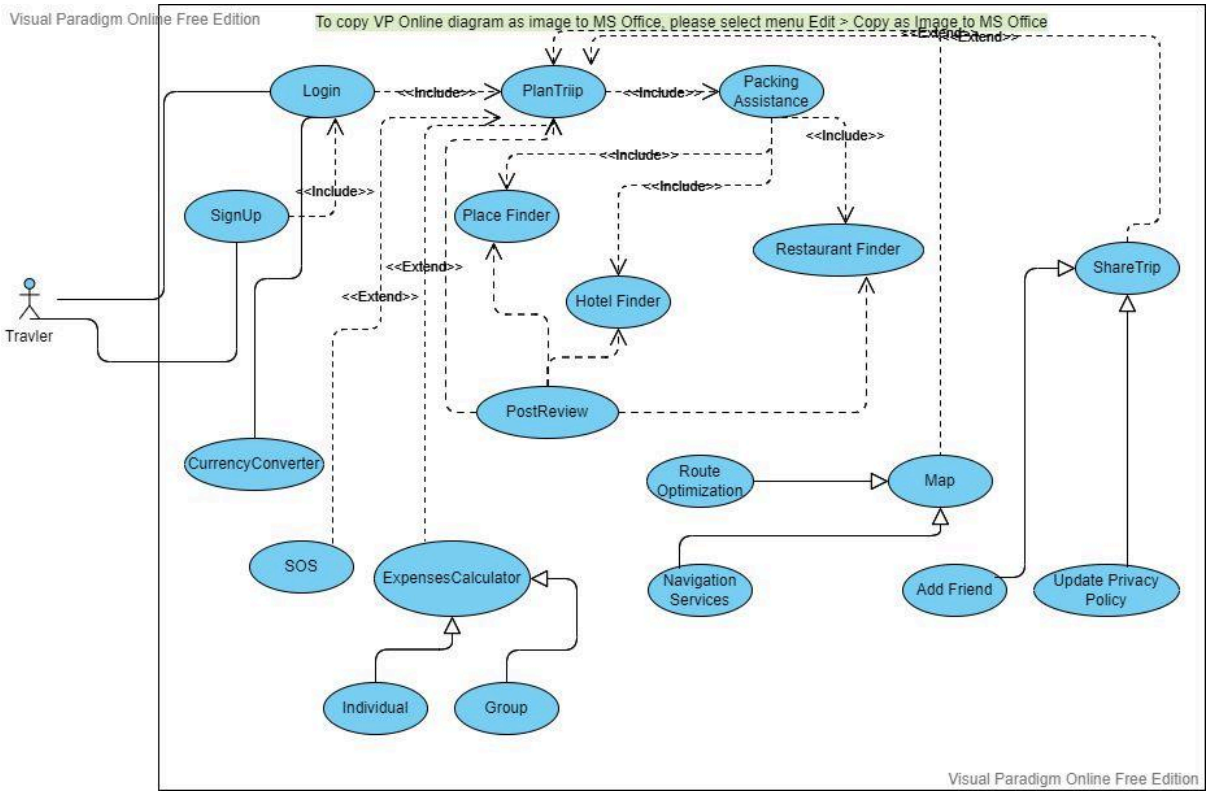
Chapter 2 - Requirement Analysis

2.1 Functional requirements

2.1.1 Functional Hierarchy



2.2 USE CASES



2.2.1 Signup

<Use case Id: Signup>		
Use case Id:	1.	
Actors:	Users (Who wish to travel)	
Feature:	<Feature from which the use case is driven>	
Pre-condition:	Correct Personal details including Email ID	
Scenarios		
Step#	**Action**	**Software Reaction**
1.	Enter First Name & Last Name	Name shall be stored in the Database
2.	Enter City	City shall be stored in the Database
3.	Enter Country	Country shall be stored in the Database
4.	Enter Email	Unique Email shall be stored in the Database
5.	Enter Password	Password shall be stored in the Database
Alternate Scenarios: Write additional, optional, branching or iterative steps. Refer to specific action number to ensure understandability.		
1a: Incase user is already signed up on the platform, the user is redirected to the sign in page. **2a: Incase user enters wrong credentials, must repeatedly be asked to enter correct credentials such as Email.**		
Post Conditions		
Step#	**Description**	

1	**User signed up**
2	**Ready to use the application**
Use Case Cross referenced	<Related use cases, which use or are used by this use case>

2.2.2 : Login

<Use case Id: Login>		
Use case Id:	2	
Actors: Users who have already signed up		
Feature: <Feature from which the use case is driven>		
Pre-condition:	Correct Credentials	
Scenarios		
Step#	**Action**	**Software Reaction**
1.	Enter Email	Checks for existing email id
2.	Enter Password	Checks for password against existing email id
Alternate Scenarios: Write additional, optional, branching or iterative steps. Refer to specific action number to ensure understandability.		
1a: If user email id does not exist, user is asked to sign up **1b: If the user password does not match, the user can change the password if forgotten.** **2a: After signup, user redirected back to login page**		
Post Conditions		
Step#	**Description**	
1	User redirected to page where he can select his travel dates	
2	Access to home page to use different features of the application	
Use Case Cross referenced	<Related use cases, which use or are used by this use case>	

2.2.3: PlanTrip

<Use case Id: PlanTrip>		
Use case Id:	3	
Actors: Traveler		
Feature: <Feature from which the use case is driven>		
Pre-condition:	User is Logged in	
Scenarios		
Step#	**Action**	**Software Reaction**
1.	Choose Destination City	Show the list of cities which user can visit
2.	Choose the start and end date of trip	Store the start and end date into a database
3.	Choose the privacy option	If public then traveler info can be shared with other and if private then the information can't be shared.
Alternate Scenarios: Write additional, optional, branching or iterative steps. Refer to specific action number to ensure understandability.		

No Alternate Scenario		
Post Conditions		
Step#	**Description**	
	Sequentially list conditions expected at the completion of the use case.	
1.	User redirected to page where he can view the essential items that he can pack for trip	
Use Case Cross referenced	Packing Assistant <use this UseCase>	

2.2.4 : Packing Assistant

<Use case Id: Packing Assistant>		
Use case Id:	4.	
Actors:	User wishing to use to find out essential items to be taken	
Feature:	Planning trip	
Pre-condition:	Location (City) where user is traveling, start and end dates	
Scenarios		
Step#	**Action**	**Software Reaction**
1.	User must know dates of traveling	Dates stored in a database
2.	User must know location of traveling	Weather shown against this location
Alternate Scenarios: Write additional, optional, branching or iterative steps. Refer to specific action number to ensure understandability.		
1a: Pick essential items of your own need from the list given by the application, track what you have taken and not.		
Post Conditions		
Step#	**Description**	
1	Check mark on items you have taken so you never forget what you missed while planning	
Use Case Cross referenced	<Related use cases, which use or are used by this use case>	

2.2.5 : Currency Convertor

<Use case Id: Currency Convertor>		
Use case Id:	5	
Actors:	Travelers who wish to check the currency conversion in order to manage their expenses	
Feature:	<Feature from which the use case is driven>	
Pre-condition:	From which currency to which currency you need to convert	
Scenarios		
Step#	**Action**	**Software Reaction**
1.	Enter From	Currency from which user wants to convert
2.	Enter To	Currency to which user wants to convert

3.	Press Convert	Press to show the results
Alternate Scenarios: Write additional, optional, branching or iterative steps. Refer to specific action number to ensure understandability.		
No Alternate Scenarios		
Post Conditions		
Step#	**Description**	
1	Correct conversion of currency from and to the currency wished by user	
Use Case Cross referenced	<Related use cases, which use or are used by this use case>	

2.2.6: ShareTrip

<Use case Id: ShareTrip>		
Use case Id:	6	
Actors:	<List of actors (external agents), indicating who initiated the use case>Travelers	
Feature:	Sharing Trip among travelers	
Pre-condition:	All traveler should be sharing a same trip	
Scenarios		
Step#	**Action**	**Software Reaction**
1.	Travelers invite other travelers.	Application will send the notification to the other traveler of an invite.
2.	Travelers can accept or reject invitations.	Application will put the accepted trip mates into the database
Alternate Scenarios: Write additional, optional, branching or iterative steps. Refer to specific action number to ensure understandability.		
No Alternate Scenario		
Post Conditions		
Step#	**Description**	
1.	**Travelers should be able to share the trip.**	
2.	**Traveler should be able to chat with each others**	
Use Case Cross referenced	Expenses<which use>	

2.2.7 : Expenses

<Use case Id: Expenses>	
Use case Id:	7
Actors:	Users keeping track of their expenses
Feature:	<Feature from which the use case is driven>
Pre-condition:	When, what, whom and how much they have spent on something

Scenarios		
Step#	**Action**	**Software Reaction**
1.	Set budget	Budget set and stored
2.	Add Expense	Expense stored depending on who paid (Self,Friend)
3.	Add Action	Storing the expense on the database
Alternate Scenarios: Write additional, optional, branching or iterative steps. Refer to specific action number to ensure understandability.		
No Alternate Scenario		
Post Conditions		
Step#	**Description**	
1	Track of Expenses	
2	Track of budget limit	
Use Case Cross referenced	<Related use cases, which use or are used by this use case>	

2.2.8: Post Review

<Use case Id: Post Review>		
Use case Id:	8	
Actors: Users who wish to post their trip review		
Feature: <Feature from which the use case is driven>		
Pre-condition:	User must have traveled or known that place	
Scenarios		
Step#	**Action**	**Software Reaction**
1.	Write the review in the text box	Review stored in database
2.	Post Action	Review uploaded
Alternate Scenarios: Write additional, optional, branching or iterative steps. Refer to specific action number to ensure understandability.		
No Alternate Scenario		
Post Conditions		
Step#	**Description**	
1	Review uploaded on the application which other users can go through	
Use Case Cross referenced	<Related use cases, which use or are used by this use case>	

2.2.9: Map & Navigation

<Use case Id: Map>

Use case Id:	9	
Actors:	Users who wish to use the Map and Routing services	
Feature:	<Feature from which the use case is driven>	
Pre-condition:	Location that need to be visited	
Scenarios		
Step#	**Action**	**Software Reaction**
1.	Enter Start Location	Start Location stored
2.	Enter Destination	Tracked to Destination from start location
Alternate Scenarios: Write additional, optional, branching or iterative steps. Refer to specific action number to ensure understandability.		
1a: **2a:**		
Post Conditions		
Step#	**Description**	
	Sequentially list conditions expected at the completion of the use case.	
Use Case Cross referenced	<Related use cases, which use or are used by this use case>	

2.2.10: SOS

<Use case Id: SOS>		
Use case Id:	10	
Actors:	Users in need of details of Emergency services	
Feature:	<Feature from which the use case is driven>	
Pre-condition:	If any emergency and want to contact SOS services	
Scenarios		
Step#	**Action**	**Software Reaction**
1.	Enter City	Services displayed as per city entered
Alternate Scenarios: Write additional, optional, branching or iterative steps. Refer to specific action number to ensure understandability.		
No Additional Scenarios		
Post Conditions		
Step#	**Description**	
	User will have SOS services contact to which he/she can contact in case of any need	

Use Case Cross referenced	<Related use cases, which use or are used by this use case>

2.2.11: Hotel Finder

<Use case Id: Hotel>		
Use case Id:	11	
Actors:	Users who wish to find the hotels in the city they are traveling	
Feature:	Plan Trip	
Pre-condition:	Plan trip to do or doing	
Scenarios		
Step#	**Action**	**Software Reaction**
1.	The city name in which the user is traveling	All hotels available in the city are fetched from the database and shown to user
2.	select Hotel for detailed info	Show detailed info of selected hotel to user
Alternate Scenarios:		
1a: **2a:**		
Post Conditions		
Step#	**Description**	
1	User will get list of all hotels	
2	Can get detailed info of hotel by selcting it	
Use Case Cross referenced	Plan Trip	

2.2.12 : Find Place

<Use case Id: Find Place>		
Use case Id:	12	
Actors:	Users who wish to search places to visit	
Feature:	<Feature from which the use case is driven>	
Pre-condition:	If need to visit places, and know places	
Scenarios		
Step#	**Action**	**Software Reaction**
1.	Enter City	City Search
2.	Select Place	Details of Place shown
Alternate Scenarios: Write additional, optional, branching or iterative steps. Refer to specific action number to ensure understandability.		
No Additional Scenarios		

Post Conditions	
Step#	**Description**
1	User will get the information regarding that place
Use Case Cross referenced	<Related use cases, which use or are used by this use case>

2.2.13: Restaurant Finder

<Use case Id: Restaurant Finder>		
Use case Id:	13	
Actors:	Traveler who wants to see popular restaurant of that city	
Feature:	PlanTrip	
Pre-condition:		
Scenarios		
Step#	**Action**	**Software Reaction**
1.	Select available Restaurant	Show all available restaurants in the city.
2.	Click the Restaurant for detail View	Show the details of the restaurant from the database.
Alternate Scenarios: Write additional, optional, branching or iterative steps. Refer to specific action number to ensure understandability.		
No Additional Scenarios		
Post Conditions		
Step#	**Description**	
1.	User can see all the restaurants that he can visit.	
Use Case Cross referenced	PlanTrip<Used by>	

2.3 Non-Functional Requirements

2.3.1 Performance Requirements

Whilst talking about the performance of the software speed should be precise enough for the user depending on the bandwidth, precision should be up to scale for the user because of traveling options, concurrency for the user is very important so for this app which will be provided, software should be reliable all the time while currency conversion, navigation services or others.

2.3.2 Safety Requirements

Any user with the valid email can be able to use this application.There will be no need for the user to give his confidential information to any entity which is related to this application.

2.3.3 Security Requirements

The data of any user can't be accessed by other users, users can share trips only if they have enabled the feature and permit the app to it. User authentication and token generation should be for permitting false logins. Encrypted hash passwords of users are stored in the database so no one can recover passwords during any sort of breach.

2.4 Features

Our mobile application offers a range of features to enhance the travel experience for tourists visiting Pakistan. These features include:

1. **Packing Assistant:** Our packing assistant feature helps users create customized packing lists based on their destination, travel dates, and planned activities. This feature provides users with a checklist of essential items they need to pack for their trip, reducing the risk of forgetting anything important. Users can also share their packing list with others and receive feedback.

2. **Hotel Recommendation:** Our application provides users with hotel recommendations based on their preferences, interests, and budget. Users can filter their search results by price, location, amenities, and other factors, making it easier to find the perfect accommodation for their trip.

3. **Trip Planner:** Our trip planner feature allows users to create a personalized itinerary for their trip, including details about places of sight, restaurants, and activities. Users can also add notes and reminders to their itinerary, making it easier to keep track of their plans.

4. **Map and Navigation Services:** Our application provides users with map and navigation services to help them navigate their way around Pakistan. Users can get directions to their destination, view points of interest, and find nearby attractions, making it easier to explore the natural beauty and historical landmarks of Pakistan.

5. **Travel Recommendations:** Our application provides users with travel recommendations based on their interests, such as adventure activities, cultural experiences, and foodie hotspots. Users can also browse through a collection of curated travel guides, providing them with inspiration and ideas for their trip.

6. **Share Trip:** Our share trip feature allows users to share their trip details with their family and friends, keeping them updated on their travels and making it easier to plan trips with others.

7. **Travel Expense Calculator:** Our application provides users with a travel expense calculator, which helps them estimate their travel costs based on their destination,

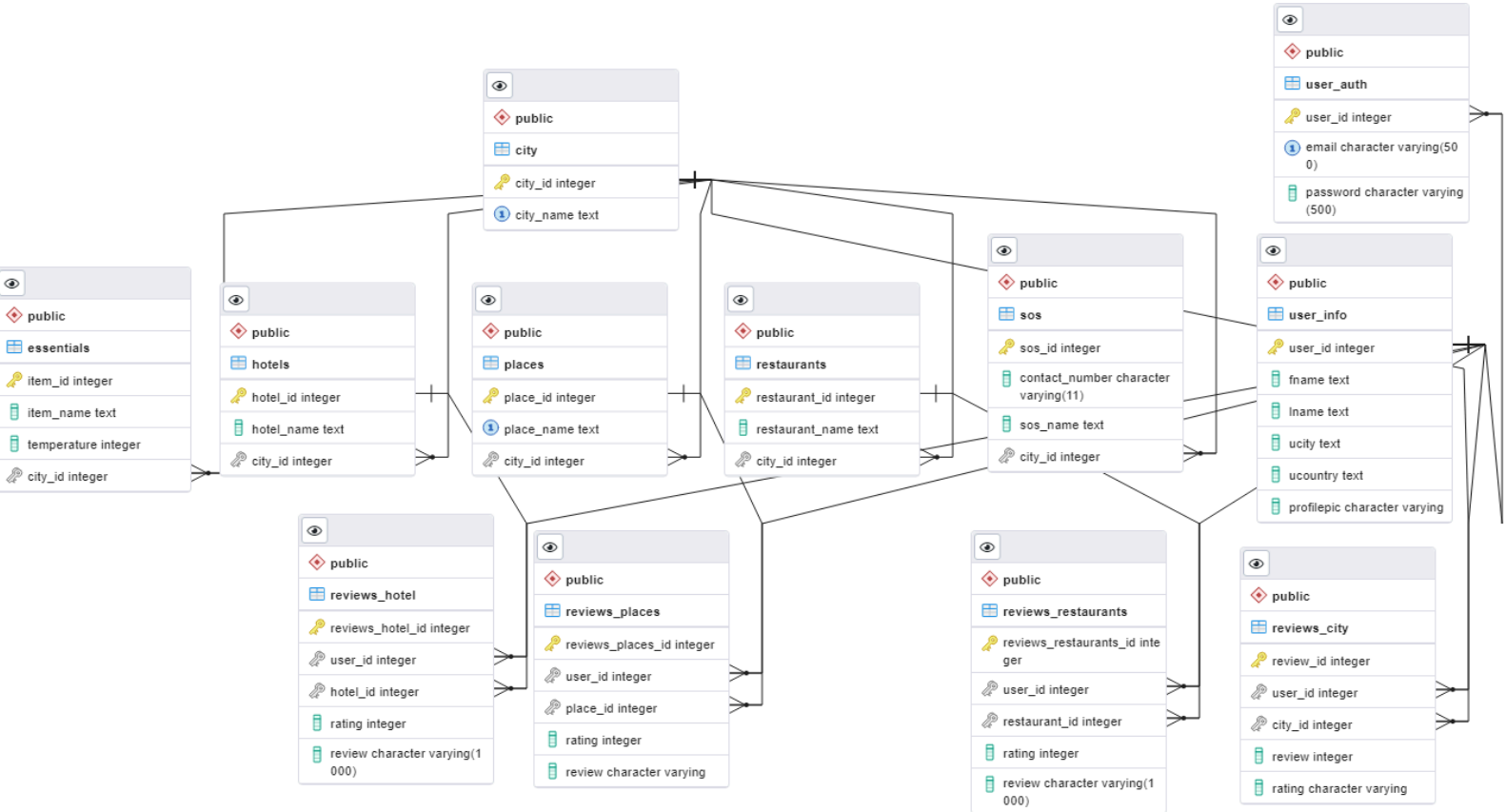
travel dates, and planned activities. This feature allows users to set a budget for their trip and keep track of their expenses, making it easier to manage their finances.

8. **Currency Conversion:** Our application provides users with a currency conversion feature, allowing them to convert currencies on the go. This feature helps users understand the local currency and make informed financial decisions during their trip.

9. **Location-Based Emergency Services:** Our application provides users with location-based emergency services, such as nearby hospitals, police stations, and embassies. This feature helps users feel more secure during their trip and provides them with quick access to emergency services if needed.

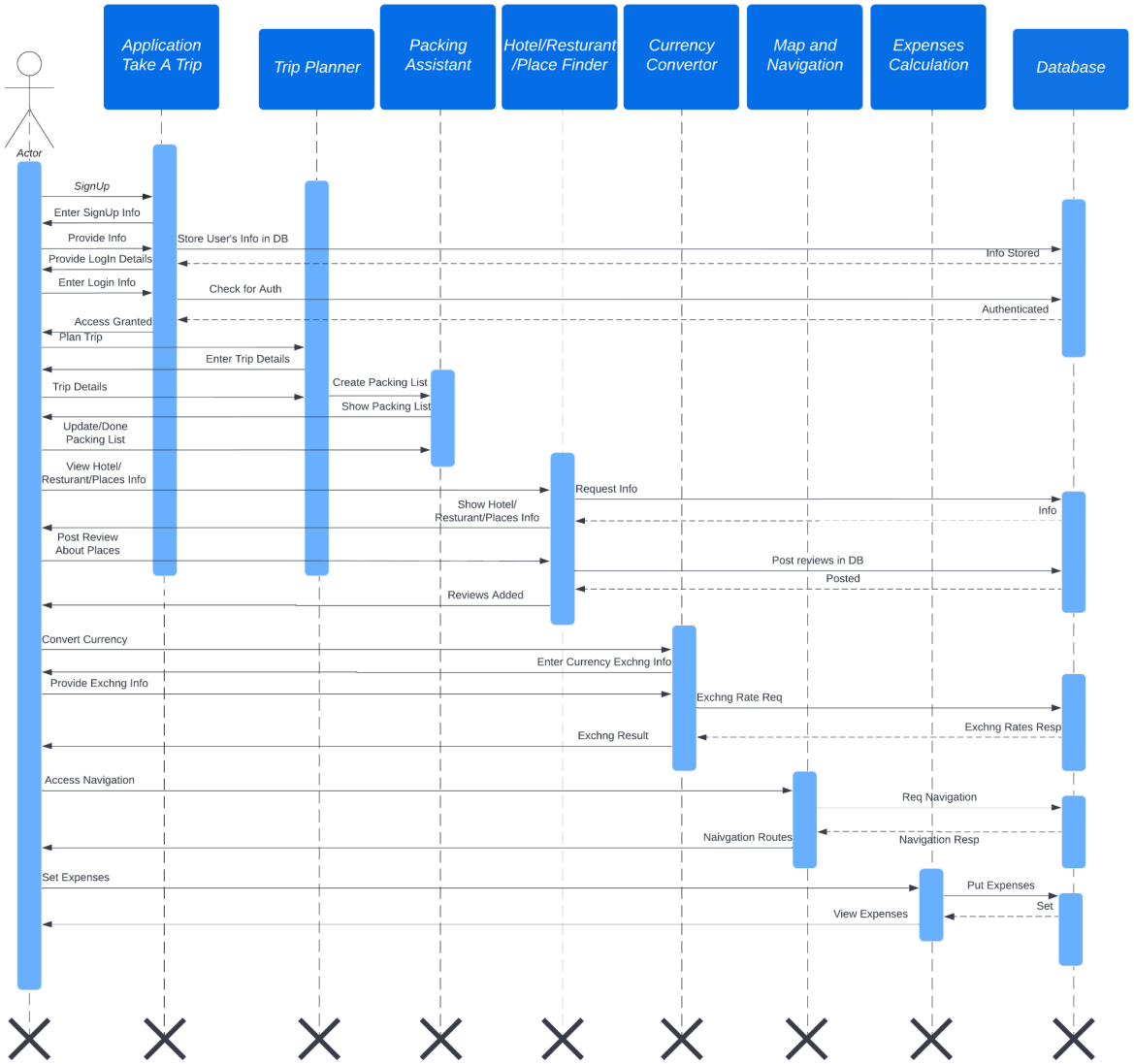
Chapter 3 - Design Detail

3.1 Database Design

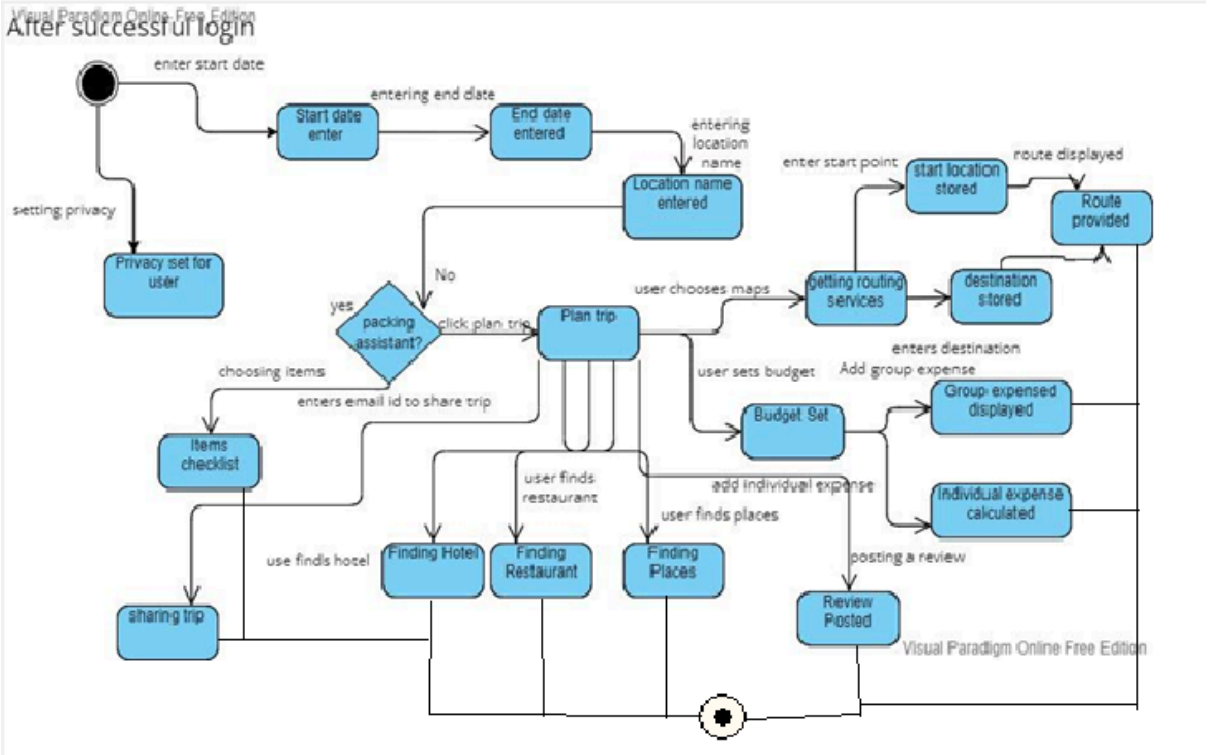


3.2 Application Design

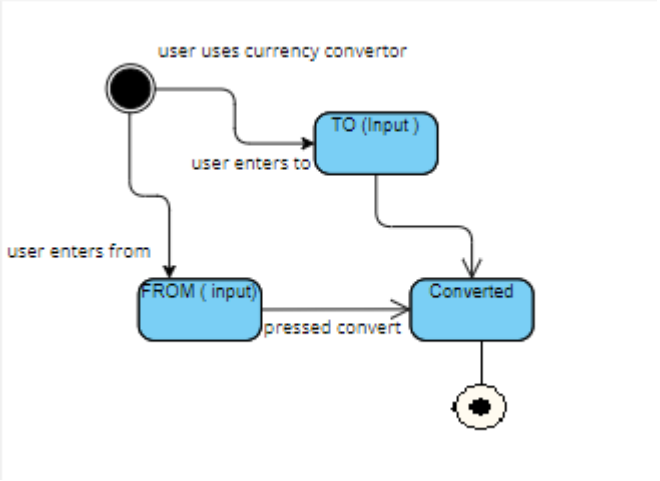
3.2.1 Sequence Diagram



3.2.2. State Diagram



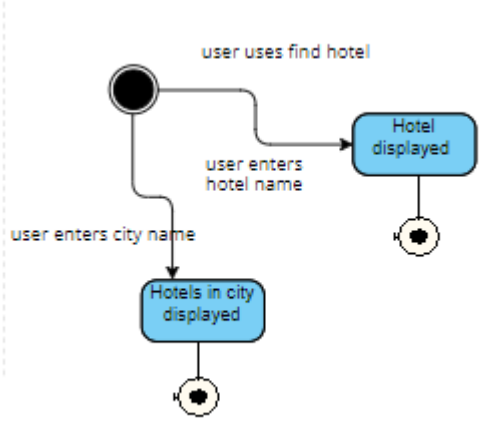
Currency Convertor



state diagram shows when the user has to convert the currency using the converter and how the state changes upon the input and button press of conversion

<Find Hotel>

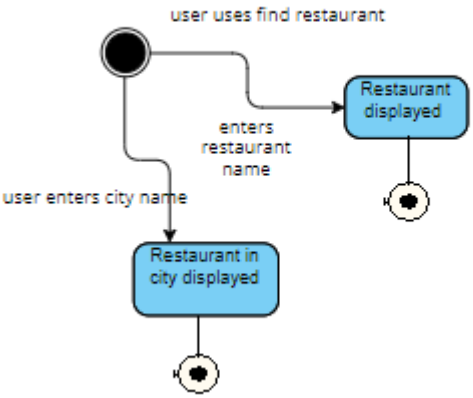
.



.

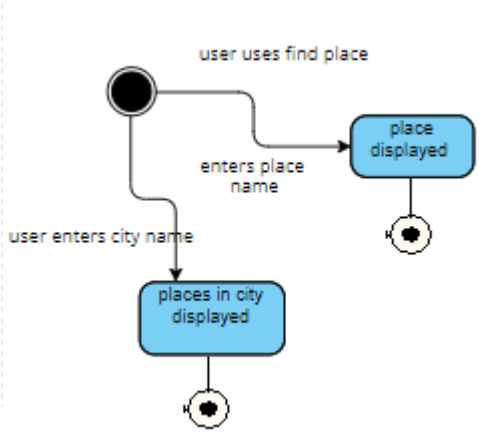
state diagram shows when the user searches for the hotels by name or city

<Find Restaurant>



state diagram shows when the user searches for the restaurants by name or city

Find Place

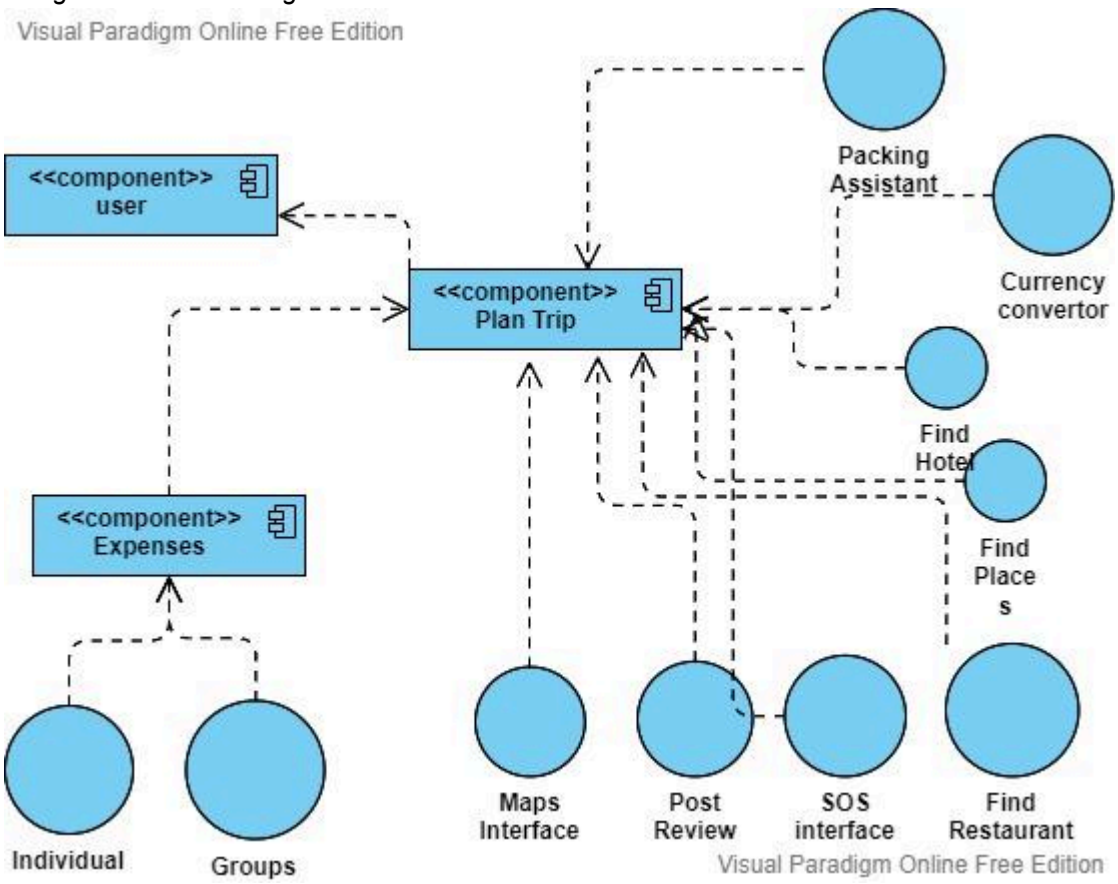


state diagram shows when the user searches for the places by name or city

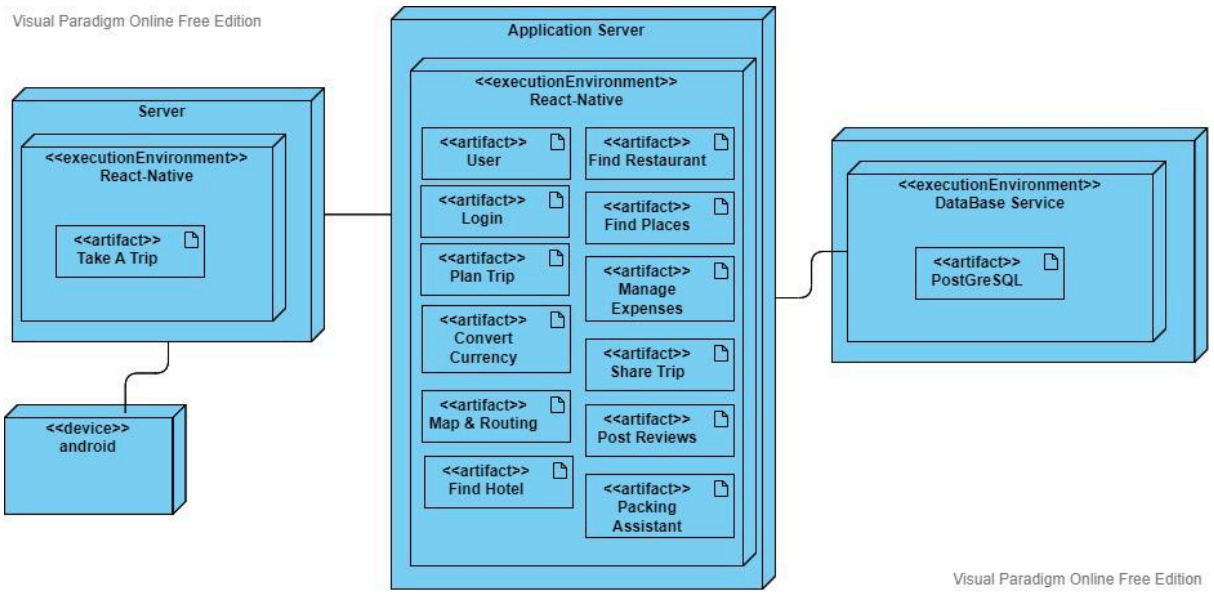
Chapter 4 - Implementation Details

4.1 System Architecture

System Designed on the user side is responsible for creating an account and logging into it. Packing assistant for users to ease their packing while traveling to a specific city. A proper management of the expenses by taking the burden of expenses alone or sharing amongst friends if traveling with them. Conversion of the currency using the Currency convertor. Posting a review where user must post about their trip in order to help other travelers to learn from their experience. A user can use Map & Navigation for the routing services.



Above component diagram shows how the manage different operations



Above Deployment Diagram shows how the whole architecture is structured of the system

4.2 Development Tools Used

- React Native FrameWork
- Node.Js
- PostGreSql
- Github
- VS Code
- Android Studio

4.3 External Hardware Interfaces

- **Gps:** User must have a device that have GPS capabilities to accurately display their location on map
- **Camera:** Allows users to take photos or upload images

4.4 Trade Offs During Implementation

During the implementation of our mobile application, we encountered several trade-offs that we had to make due to limitations in time, resources, and technical expertise. These trade-offs are important to note, as they represent the compromises we had to make during the development of our application. Some of the trade-offs we encountered are:

1. **Quality vs Quantity of Features:** As an academic project, we had limited time and resources to develop our mobile application. Therefore, we had to prioritize the features we wanted to include in our application. While we aimed to include as many features as possible, we had to balance the quality of each feature with the number of features we could realistically include.

2. **User Interface vs Functionality:** We aimed to create a user-friendly and visually appealing user interface for our mobile application. However, we had to ensure that the functionality of our application was not compromised for the sake of a more aesthetic interface. Therefore, we had to strike a balance between the user interface and the functionality of our application.

3. **Compatibility vs Optimization:** As a cross-platform application, we had to ensure that our application was compatible with both iOS and Android devices. However, this meant that we had to sacrifice some degree of optimization for each platform to achieve compatibility. Therefore, we had to strike a balance between compatibility and optimization.

4. **Security vs Convenience:** We aimed to provide our users with a secure and safe application that protected their personal data. However, this meant that we had to implement security measures that might compromise the convenience of our application. Therefore, we had to strike a balance between security and convenience.

Chapter 5 - Testing

5.1 Introduction:

Testing is an essential part of software development that ensures the quality and reliability of the app. By identifying and fixing defects and issues during the testing phase, you can reduce the risk of bugs and crashes in the live app, which can lead to negative user experiences and reputational damage. In this chapter, we will cover the different types of testing that were performed for the "Take A Trip" app, including validation testing, integration testing, and user acceptance testing.

5.2 Environment Needs:

For testing your "Take A Trip" app, we set up an environment that included the following components:

Emulators and physical devices: To test the app on different platforms and devices, we used a combination of emulators and physical devices that covered the most popular iOS and Android configurations.
Testing tools: We used a variety of testing tools to automate tests and track results, including Appium for functional testing, XCUITest and Espresso for UI testing, and Calabash for cross-platform testing.
Test data: To simulate realistic user behavior and usage, we generated test data that included a variety of locations, points of interest, and travel itineraries.

Staging environment: We set up a staging environment that mirrored the production environment to test app updates and new features before releasing them to production.

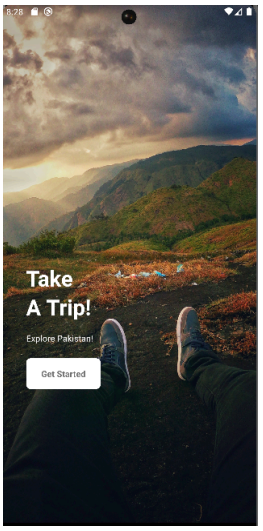
5.3 Validation Testing:

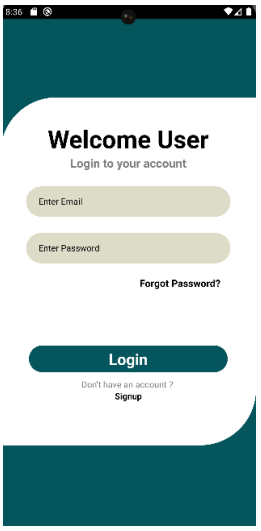
For the validation testing of the "Take A Trip" app, we performed the following tests:

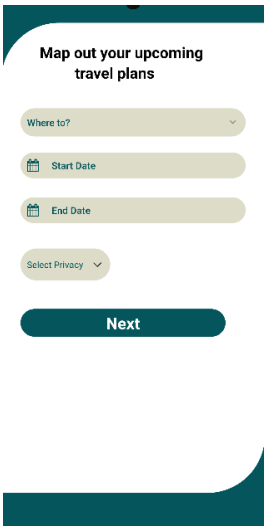
Functional testing: We verified that the app's features and functionality worked as intended, including searching for destinations, displaying information about points of interest, and generating travel itineraries. We used Appium to automate functional tests and verify the expected behavior of the app.

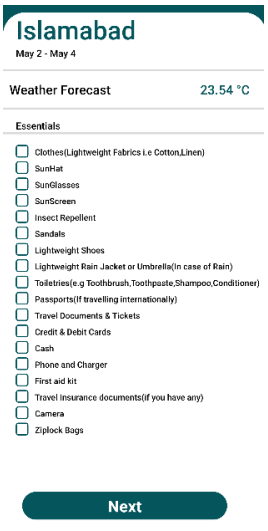
Usability testing: We evaluated the app's user interface and user experience to ensure that it was intuitive and easy to use

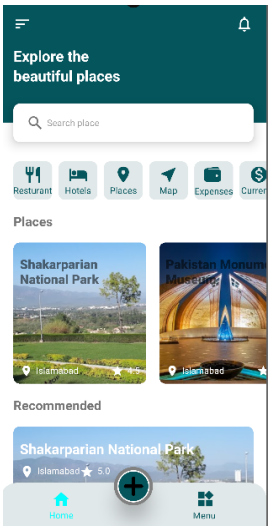
Chapter 6 - Output

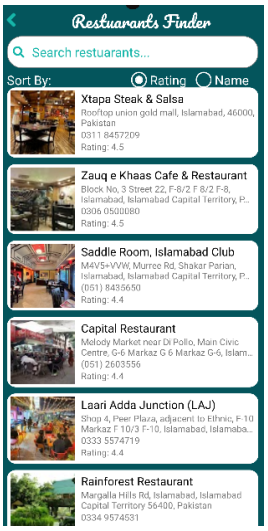


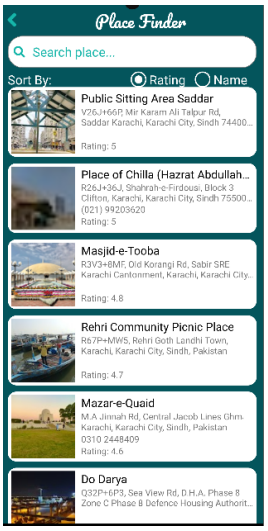


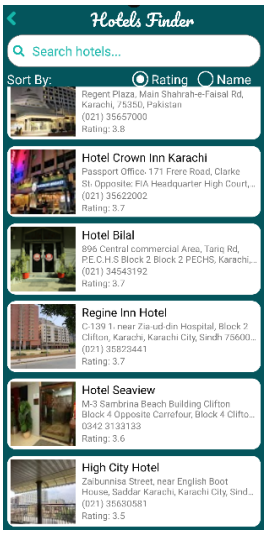


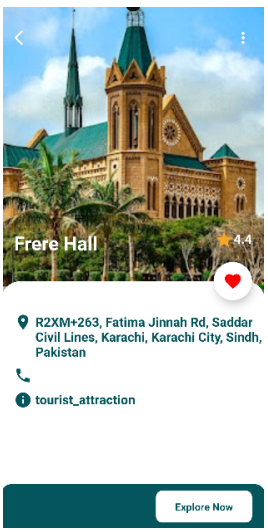


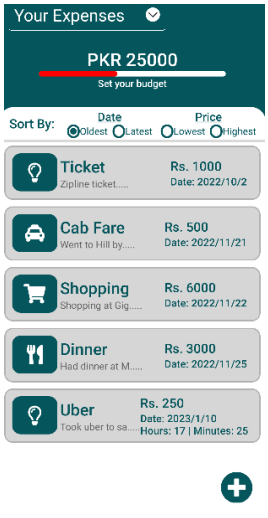


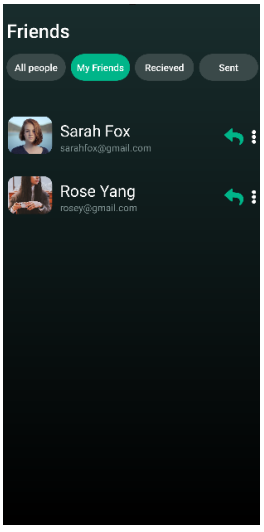


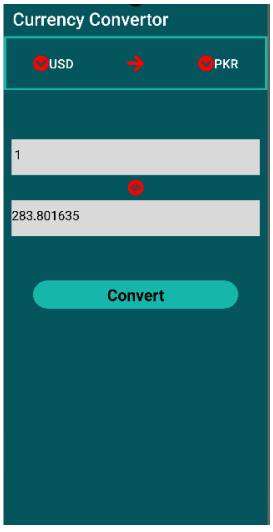


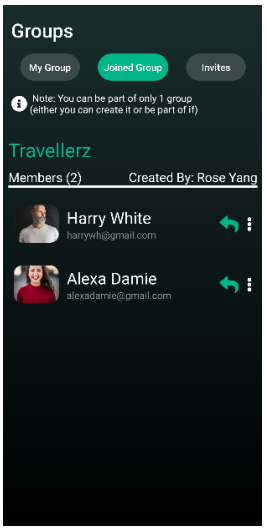






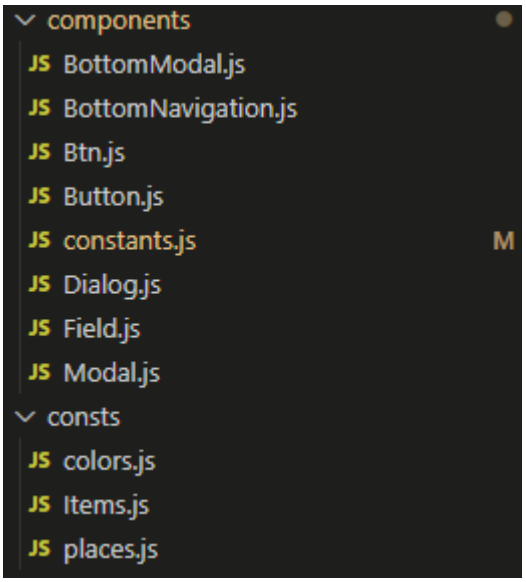


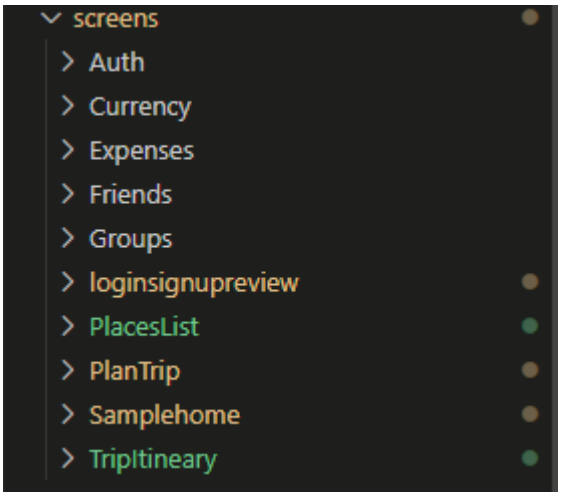


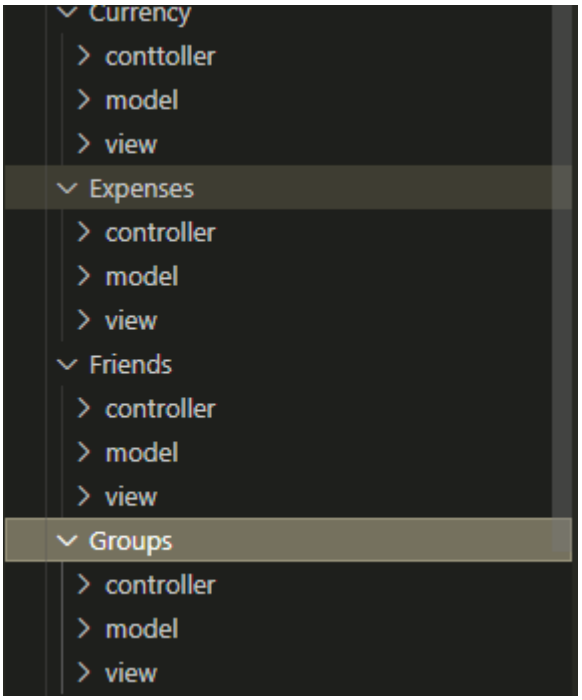


Chapter 7- Code

As React Native is used for frontend and NodeJS for backend so language used is JavaScript.
As the architectural pattern used is MVC so the file structure is also designed accordingly to it.







CODE for LOGIN:
import axios from 'axios';

```
import React, { useEffect } from 'react';
import { useState } from 'react';
import { View, StyleSheet, Text, TouchableOpacity, ScrollView, Alert } from 'react-native';
import Background from './Background';
import Btn from '../../components/Btn';
import Field from '../../components/Field';
import COLORS from '../../consts/colors';
import { localHost, screenHeight } from '../../components/constants';
import {NavigationContainer} from '@react-navigation/native';
//DataBase

const Login = (props) => {
    //HOOKS FOR VALIDATIOn
    const [valueofemail, onChangeEmail] = useState('');
    const [emailerror, SetEmailError] = useState('');
    const [valueofpassword, onChangePassword] = useState('');
    const [passworderror, SetPasswordError] = useState('');
    //VALIDATION FUNCTION

    function validation(e, p) {
        let flag1 = false;
        let flag2 = false;
        if (e == '') { flag1=false;SetEmailError('Please Enter Email'); }
        else {
            let rjx = new RegExp('[a-z0-9]+@[a-z]+\.[a-z]{2,3}');
            if (!rjx.test(e)) {
                SetEmailError('Enter Correct Format Of Email');
            }
            else {
                SetEmailError('');
                flag1 = true;
            }
        }
        if (p == '') { flag2=false; SetPasswordError('Please Enter Passwword'); }
        else {
            if (p.length < 6) { SetPasswordError('Minimum Password Length required is 5'); }
            else {
                SetPasswordError('');
                flag2 = true;

            }
        }
        if (flag1 == true && flag2 == true) {
        }
        //AXIOS LOGIN FUNCTION

axios.get(`http://${localHost}/user_info/${e.toString()}/${p.toString()}`).then(function(response
){
```

```
        console.log(response.data)

        if(response.data==false)
        {
            Alert.alert(
                'Incorrect Credentials'
            );

        }
      if (response.data[0].pass==p.toString())
   {
      Alert.alert(
          'Successful Login'
      );
      id_user=response.data[0].user_id
      SetEmailError('');
      SetPasswordError('');
      props.navigation.replace("PlanTrip",{id_user:id_user})
      e="";
      p="";
   }else{
      // console.log(response.data[0])
      // Alert.alert(
      //     'Incorrect Credentials'
      // );
}}).catch(function(error)
   {})
}
   ///PAGE CODE
   return (
         <View style={{backgroundColor:COLORS.primary, alignItems: "center", width:
'100%',height:'100%' }}>

               <ScrollView showsVerticalScrollIndicator={false} style={{
                height: '80%', width: '100%'}} >
               <View style={{
                  backgroundColor: "white", height: '80%', width:
'100%',marginBottom:screenHeight*0.1,
                  borderTopLeftRadius: 90,borderBottomRightRadius:90,paddingTop: 40,
alignItems: 'center', marginTop:screenHeight*0.15}}>
                  <Text style={{ fontSize: 40, color: 'black', fontWeight: "bold" }}>Welcome
User</Text>
                  <Text style={{ color: "grey", fontSize: 19, fontWeight: "bold", marginBottom:
20 }}>Login to your account</Text>
                  <Field placeholder="Enter Email" keyboardType={"email-address"}
value={valueofemail} onChangeText={text => onChangeEmail(text)} />
                  <Text style={{ color: 'red', fontSize: 12, fontWeight:'bold',marginRight:220
}}>{emailerror}</Text>
```

```
                <Field placeholder="Enter Password" secureTextEntry={true}
value={valueofpassword}
                    onChangeText={text => onChangePassword(text)} />
                <Text style={{ color: 'red', fontSize: 12, fontWeight:'bold',marginRight:200
}}>{passworderror}</Text>
                <View style={{ alignItems: 'flex-end', width: '80%', paddingRight: 16,
marginBottom: 80 }}>
                    <Text style={{ color: 'black', fontWeight: 'bold', fontSize: 16 }}>Forgot
Password?</Text>
                </View>
                <Btn textColor='white' bgColor={COLORS.primary} btnLabel="Login"
Press={() => { validation(valueofemail, valueofpassword) }} />
                    <Text>Don't have an account ? </Text>
                    <TouchableOpacity onPress={() =>
{SetEmailError('');SetPasswordError('');props.navigation.navigate("Signup");}}>
                        <Text style={{ color: 'black', fontWeight: 'bold' }}>Signup</Text>
                    </TouchableOpacity>
            </View>
            </ScrollView>
        </View>
    );
}
const styles = StyleSheet.create({})

export default Login;
```

Chapter 8 - Limitations

While the "Take A Trip" travel guide app has been designed to provide users with an easy and intuitive way to plan and navigate their trips, there are several limitations to the app that should be noted.

- **Data Availability:**
  The "Take A Trip" app relies on various data sources to provide users with information on points of interest, travel routes, and other relevant details. However, the availability and accuracy of this data may vary depending on the location and source. In some cases, the app may not have access to the latest or most comprehensive data, which could lead to outdated or incomplete information.

- **Network Connectivity:**
  To provide users with real-time information and updates, the "Take A Trip" app requires a stable internet connection. However, in areas with poor network connectivity or limited access to Wi-Fi, users may experience slower loading times or difficulty accessing certain features of the app.

- **User Input:**
  The accuracy and completeness of the information provided by the "Take A Trip" app also depend on the quality and completeness of user input. For example, if a user enters incomplete or inaccurate information when planning their trip, the app may provide incomplete or inaccurate recommendations.

Chapter 9 - Future Work

While the "Take A Trip" travel guide app has been developed to provide users with an intuitive and comprehensive way to plan and navigate their trips, there are several areas where additional development and enhancements could be made to improve the user experience and functionality of the app.

- **Offline Functionality:**
  To address concerns around network connectivity and accessibility in areas with poor internet access, future development of the "Take A Trip" app could focus on implementing offline functionality. This would allow users to access key features of the app, such as travel itineraries and point of interest information, even when they do not have an internet connection.

- **Personalization:**

While the app does allow users to input some basic preferences and information when planning their trips, future development could focus on providing more personalized recommendations and suggestions. This could involve leveraging machine learning and AI algorithms to analyze user behavior and preferences to provide more tailored recommendations for points of interest, travel routes, and other aspects of trip planning.

- **Integration with Social Media:**
  To further enhance the social aspect of travel planning and sharing, future development of the "Take A Trip" app could include integration with social media platforms. This could allow users to easily share their travel plans and experiences with friends and family, as well as access recommendations and tips from other users in their social networks.

Chapter 10 - Conclusion

In conclusion, the development of the "Take A Trip" travel guide app represents a significant step forward in the world of travel planning and navigation. By leveraging the latest mobile technologies and data sources, we have created an app that provides users with an easy and intuitive way to plan and navigate their trips, regardless of their level of experience or familiarity with the destination.

Throughout the development process, we have been committed to ensuring that the "Take A Trip" app is not only functional and comprehensive but also user-friendly and accessible. By incorporating user feedback and suggestions, we have been able to refine and enhance the app's features and functionality to better meet the needs of our users.

While the "Take A Trip" app is not without its limitations, we are confident that it represents a significant improvement over existing travel planning and navigation tools. Moving forward, we will continue to invest in the development and enhancement of the app to further improve its functionality, accuracy, and usability.

We would like to express our gratitude to our development team and supervisor for their contributions to this project, as well as to our users for their feedback and support. We are excited about the future of the "Take A Trip" app and look forward to continuing to serve as the go-to travel planning and navigation tool for users.