



## American International University-Bangladesh (AIUB)

### Department of Computer Science

### Faculty of Science & Technology (FST)

### Tripmaker A Travel Agency Management System

A Software Quality and Testing Project Submitted  
By

Semester: Spring_24_25				Section: E	Group No: 01
SL	SN	Student Name	Student ID	Individual Contribution (in %)	Total Marks: 50
					Earned Marks:
A	23	SHA MOHAMAD YEAHIA IDRIS	22-46787-1	100%	
B	24	DIP ACHORJEE SHOKAL	22-46788-1	100%	
C	25	S M ABID HASAN	22-46789-1	100%	
D	26	MD. REZWAN MUJAHID RUDRO	22-46802-1	100%	

The project will be Evaluated for the following Course Outcomes

EVALUATION CRITERIA	Total Marks (50)	
Revision History, Test Plan Identifier, Reference Materials, Problem Background, Solutions	[10 Marks]	
Requirements Specification (System feature, Quality Attributes, System Interface, Project Requirements)	[10 Marks]	
Item Not to be tested, Testing approach (Testing levels, tools, meetings), Test cases	[10 Marks]	
Item pass/fail criteria, Test deliverables, Staffing and Training, Responsibilities, Scheduling, Risk	[10 Marks]	
Approval, Format, Submission, and Defense	[10 Marks]	

---

# Software Test Plan

for

## Tripmaker: A Travel Agency Management System

Version 1.0 approved

Prepared by Sha Mohammad Yeahia Idris, Dip Achorjee Sokal, S M Abid Hassan,  
Md. Rezwan Mujahid Rudro

American International University-Bangladesh

10-Jun-2025

## Table of Contents

<b>Revision History .....</b>	<b>3</b>
<b>1. TEST PLAN IDENTIFIER: TPM-TP-01.25.....</b>	<b>4</b>
<b>2. REFERENCE MATERIALS.....</b>	<b>4</b>
<b>3. INTRODUCTION.....</b>	<b>4</b>
3.1 Background to the Problem.....	4
3.2 Solution to the Problem.....	5
<b>4. REQUIREMENT SPECIFICATION .....</b>	<b>5</b>
4.1 System Features.....	5
4.2 System Quality Attributes.....	8
4.3 System Interface .....	10
4.4 Project Requirements.....	19
<b>5. FEATURES NOT TO BE TESTED.....</b>	<b>24</b>
<b>6. TESTING APPROACH .....</b>	<b>25</b>
6.1 Testing Levels .....	25
6.2 Test Tools .....	27
6.3 Meetings .....	28
<b>7. TEST CASES/TEST ITEMS .....</b>	<b>29</b>
<b>8. ITEM PASS/FAIL CRITERIA .....</b>	<b>39</b>
<b>9. TEST DELIVERABLES .....</b>	<b>40</b>
<b>10. STAFFING AND TRAINING NEEDS .....</b>	<b>41</b>
<b>11. RESPONSIBILITIES .....</b>	<b>42</b>
<b>12. TESTING SCHEDULE.....</b>	<b>43</b>
<b>13. PLANNING RISKS AND CONTINGENCIES .....</b>	<b>43</b>
<b>14. APPROVALS.....</b>	<b>46</b>

## Revision History

Revision	Date	Updated by	Update Comments
0.1	2025.05.01	SHA MOHAMAD YEAHIA IDRIS	First Draft
0.2	2025.05.08	DIP ACHORJEE SHOKAL	Second Draft
0.3	2025.05.15	S M ABID HASAN	Third Draft
0.4	2025.05.22	MD. REZWAN MUJAHID RUDRO	Fourth Draft
0.5	2025.06.15	SHA MOHAMAD YEAHIA IDRIS	Final Draft

## 1. TEST PLAN IDENTIFIER: **TPM-TP-01.25**

## 2. REFERENCE MATERIALS

Below are key documents and sources referenced in preparing this test plan. These include both formal project documentation and external reviews/articles that highlight real-world issues and user expectations:

1. **Software Requirement Specification (SRS)** – TripMaker Project
2. **PhocusWire article:**  
PhocusWire. (2023, November 9). *Travel apps have work to do in improving customer satisfaction*. PhocusWire. <https://www.phocuswire.com/consumer-online-travel-booking-behavior>
3. **Travelport 2024 Report (via Phocuswright):**  
Travelport. (2024). *Global traveler research 2024: The new era of choice*. Phocuswright. <https://www.phocuswright.com/Travel-Research/Research-Updates/2024/Travelport-Global-Traveler-Report-2024>

## 3. INTRODUCTION

### 3.1 Background to the Problem

In the modern travel industry, managing bookings for transportation, accommodation, and activities often requires interacting with multiple platforms, leading to fragmented data and inefficient manual processes. Travel agencies struggle to coordinate services like flights, buses, trains, hotels, and tour activities in a seamless manner. Traditionally, travelers must visit several websites or apps to book transportation, reserve hotels, and plan activities like dining or adventure sports. This scattered process causes confusion, wastes time, and increases the risk of booking errors or missed reservations.

The root of this problem lies in the absence of a unified platform that brings together all essential travel-related services. Users are forced to manage multiple accounts, compare services manually, check availability across various sources, and track different confirmation emails. On the agency side, the lack of centralized control leads to greater administrative overhead and poor customer service.

This issue is crucial because it directly impacts customer satisfaction, business efficiency, and service reliability. In a highly competitive tourism market, addressing this problem can enhance service quality, simplify operations, and strengthen customer relationships.

### 3.2 Solution to the Problem

The proposed solution is **TripMaker** - a unified travel agency management system that enables customers to book transportation, hotels, and activities in one place. Designed for both customers and admins, it facilitates smooth interaction between users and service providers.

TripMaker combines all travel planning features into a single, user-friendly application. Customers can register, search services, book, view their history, and update profiles, while admins manage services and user data through a centralized dashboard. This integration removes the need for multiple platforms and enhances accuracy, convenience, and efficiency. TripMaker supports the business objective of improving customer satisfaction, reducing operational overhead, and boosting revenue through a streamlined experience.

While platforms like **Booking.com**, **TripAdvisor**, **Expedia**, and **MakeMyTrip** offer similar services, they typically focus on individual features and lack integrated admin-user interaction. In many cases, activity booking is not fully integrated or tailored to the customer's journey.

TripMaker stands out by offering a tightly coupled admin-customer system where both parties interact through a shared database and real-time service updates. Unlike existing solutions that provide partial support, TripMaker delivers a complete travel planning ecosystem ideal for local and agency-level deployment.

## 4. REQUIREMENT SPECIFICATION

### 4.1 System Features

#### 1. Login

##### Functional Requirements

- 1.1 The system shall allow users to log in using their username and password.
- 1.2 The system shall verify login credentials against stored data.
- 1.3 If login fails, the system shall display an error message with an option to reset the password via email verification.

Priority Level: High

Precondition: User must provide correct username and password.

Cross reference: 2.1, 4.1, 10.1, 11.1

#### 2. SignUp

##### Functional Requirements

- 2.1 The system shall allow users to sign up by providing required fields: name, username, email, phone number, gender, password, confirm password, and address.
- 2.2 The system shall validate password strength (minimum 8 characters, including digits and

special characters).

2.3 The system shall verify the uniqueness of usernames.

Priority Level: High

Precondition: User must provide all valid information.

Cross reference: 1.1, 4.1, 5.2

### **3. Logout**

#### **Functional Requirements**

3.1 The system shall allow users to log out via a logout button after confirmation.

3.2 Upon logout, the user will be redirected to the home page.

Priority Level: Low

Precondition: User must be logged in.

Cross reference: 1.1, 5.2, 4.1

### **4. Profile**

#### **Functional Requirements**

4.1 The system shall allow users to view and update profile information.

4.2 The system shall allow optional profile image uploads.

Priority Level: Medium

Precondition: User must be logged in.

Cross reference: 1.1, 2.1, 5.2

### **5. Home Page**

#### **Functional Requirements**

5.1 The home page shall display service information, branding, search bar, menu bar, and featured deals.

5.2 The page shall dynamically change based on user login status (sign-in/sign-out/profile visibility).

5.3 It shall include destination ratings and contact information.

Priority Level: High

Precondition: User has access to the homepage.

Cross reference: 6.1, 7.1, 8.1, 9.1, 3.2

### **6. Transportation Booking**

#### **Functional Requirements**

6.1 The system shall allow users to select a transport category (bus, train, flight), from/to

locations, and journey date.

6.2 The system shall return transport options if required fields are filled.

6.3 The system shall display available seats and allow bookings.

Priority Level: Medium

Precondition: User must access homepage.

Cross reference: 5.1, 10.1, 11.1

## **7. Hotel Booking**

### **Functional Requirements**

7.1 The system shall allow hotel search based on location and check-in/check-out dates.

7.2 The system shall show hotel information, available rooms, and allow users to book.

7.3 Additional services (meals) can be added during booking.

Priority Level: Medium

Precondition: User must access homepage.

Cross reference: 6.1, 8.1, 10.1, 11.1

## **8. Activities**

### **Functional Requirements**

8.1 The system shall suggest activities based on the hotel location selected.

8.2 The system shall allow users to book optional services like dining and adventure sports.

Priority Level: Medium

Precondition: User must access homepage.

Cross reference: 7.1, 9.1, 10.1, 11.1

## **9. Packages**

### **Functional Requirements**

9.1 The system shall display combo packages that include multiple services.

9.2 Users shall be able to book all-in-one packages from this page.

Priority Level: Medium

Precondition: User must access homepage.

Cross reference: 6.1, 7.2, 8.2, 10.1, 11.1

## **10. Booking Info**

### **Functional Requirements**

10.1 Users shall be able to view, add, or delete their previous bookings.

Priority Level: High

Precondition: User must be logged in.

Cross reference: 6.3, 7.2, 8.2, 9.2, 1.1, 4.1

## 11. Wallet

### Functional Requirements

11.1 The wallet shall show a summary of booking payments and generate receipts.

11.2 The system shall support cash and online payment methods.

11.3 Bookings may be cancelled if payment is not made within a specified time.

Priority Level: High

Precondition: User must be logged in.

Cross reference: 6.3, 7.3, 8.2, 9.2, 10.1

## 12. About Page

### Functional Requirements

12.1 The About page shall provide policy details, company info, achievements, and contact.

Priority Level: Low

Precondition: User must access homepage.

Cross reference: 5.3, 6, 7, 8, 9, 10, 11

## 4.2 System Quality Attributes

**QA1 – Usability:** A user shall be able to complete the sign-up process, update their profile, and make a transport or hotel booking within five minutes under normal system conditions. The interface shall guide the user with clear validation messages and intuitive navigation.

Priority Level: High

Precondition: User has internet access and accesses the system via a supported device/browser.

Cross reference: QA2, QA4

**QA2 – Performance:** The system shall load critical pages (e.g., login, home, search results, profile) within 3–5 seconds under normal load of up to 100 concurrent users.

Priority Level: High

Precondition: Server load is within standard operational thresholds ( $\leq 100$  concurrent users), and there are no network outages.

Cross reference: QA1, QA5

**QA3 – Security:** All user passwords shall be encrypted using secure salted hashing algorithms. Online payments must be protected using SSL(Secure Sockets Layer) encryption and verified payment gateways. The system shall lock user accounts after five failed login attempts.

Priority Level: High



Precondition: User is attempting login or initiating a payment transaction.

Cross reference: QA2, QA6

**QA4 – Accessibility:** The system shall be accessible to users with visual and motor impairments by supporting screen readers, keyboard navigation, and high-contrast display modes.

Priority Level: Medium

Precondition: User accesses the system through an assistive device or software.

Cross reference: QA1, QA6

**QA5 – Scalability:** The system shall support up to 1000 concurrent users without significant performance degradation by implementing horizontal scaling, database indexing, and caching strategies.

Priority Level: Medium

Precondition: The system is deployed on scalable infrastructure (e.g., cloud) with proper resource allocation.

Cross reference: QA2, QA6

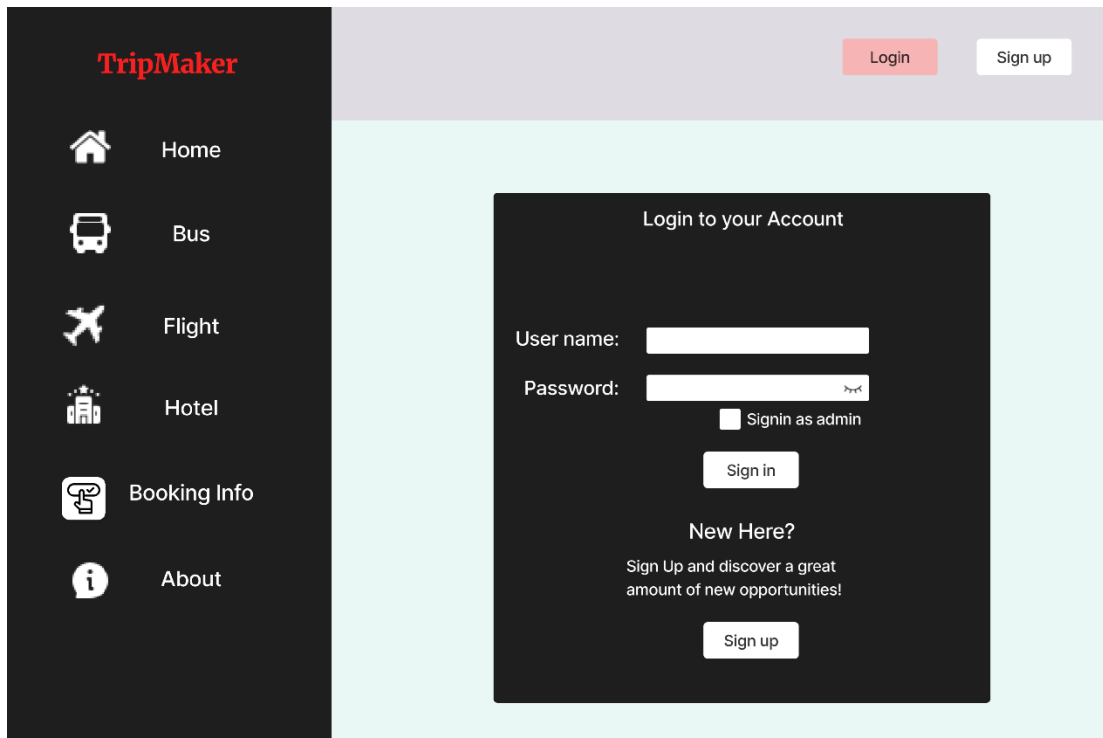
**QA6 – Compatibility:** The system shall function consistently across all major browsers (Chrome, Firefox, Safari, Edge) and mobile platforms, maintaining layout, functionality, and security across device types.

Priority Level: Medium

Precondition: User accesses the system through an updated and supported browser or mobile OS.

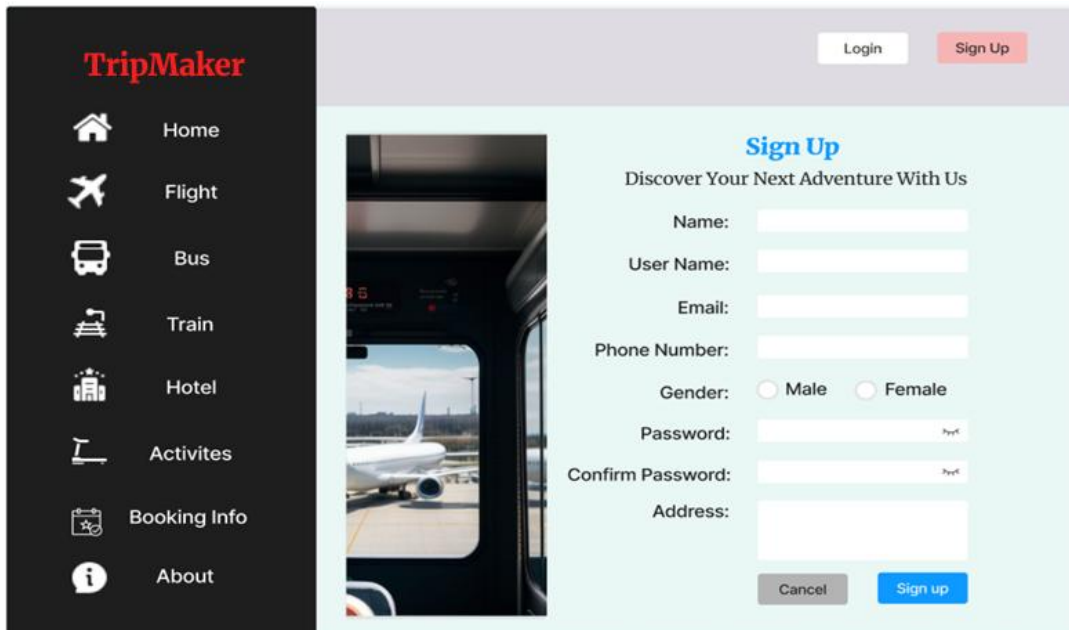
Cross reference: QA3, QA4

### 4.3 System Interface

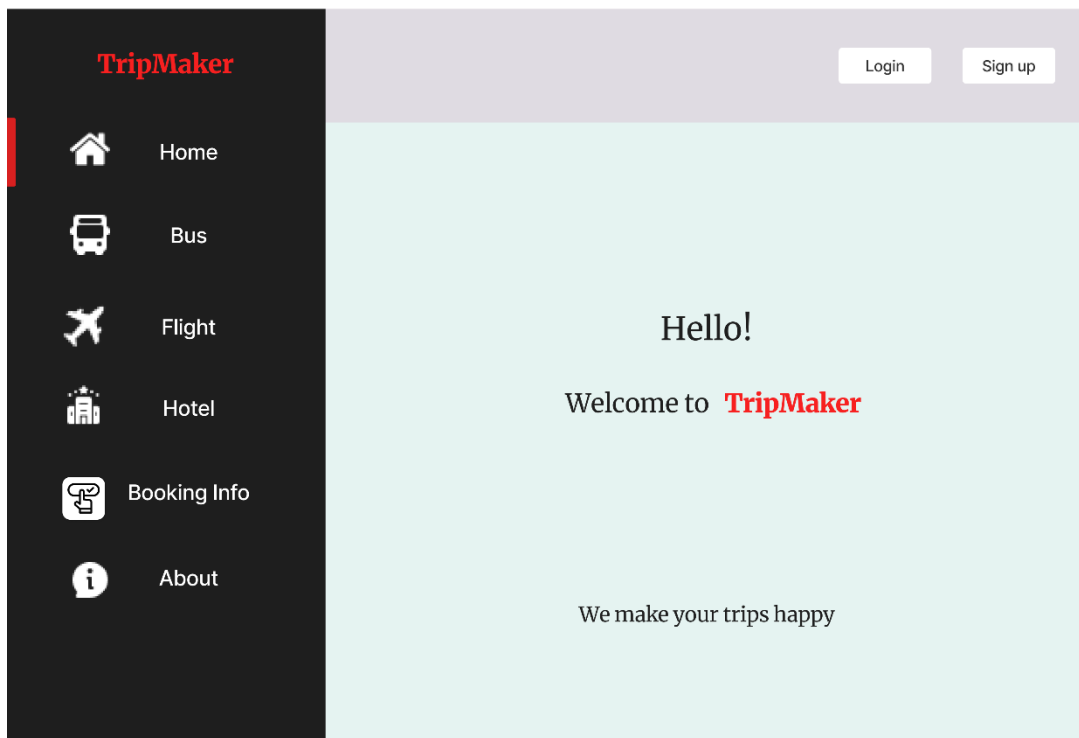
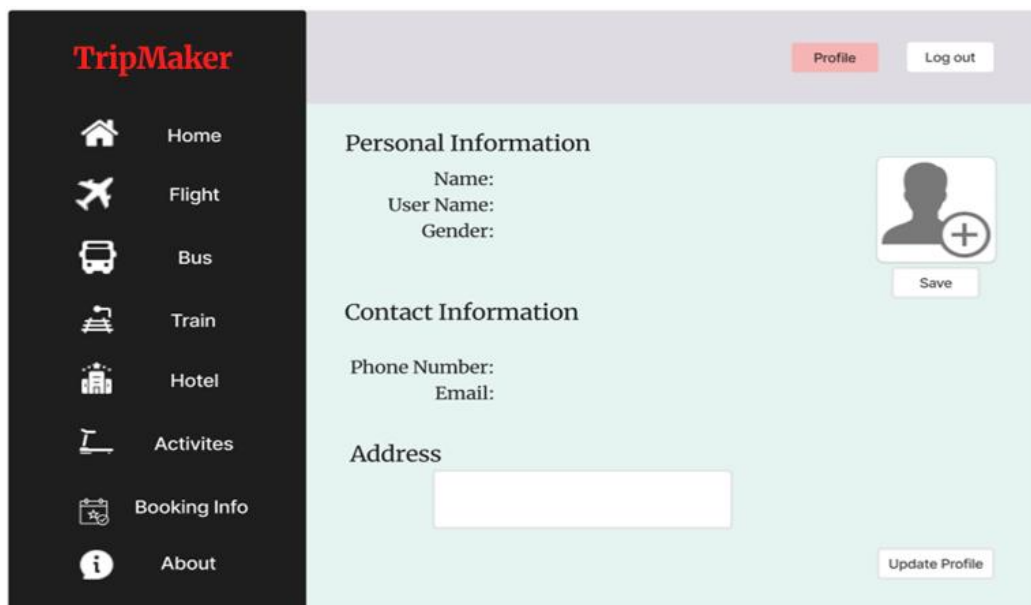


The screenshot displays the TripMaker login interface. On the left, a dark sidebar contains the TripMaker logo and navigation links: Home, Bus, Flight, Hotel, Booking Info, and About. The main content area features a light blue background with a central dark grey login box. At the top right of the page, there are 'Login' and 'Sign up' buttons. The login box is titled 'Login to your Account' and includes fields for 'User name:' and 'Password:'. A checkbox for 'Signin as admin' is located below the password field. A 'Sign in' button is positioned below the checkbox. Below the login box, a 'New Here?' section encourages users to sign up, stating 'Sign Up and discover a great amount of new opportunities!' with a 'Sign up' button.

Figure 1: Login Page



The screenshot displays the TripMaker sign-up interface. On the left, the same dark sidebar as the login page is present, but with an additional 'Activites' link. The main content area has a light blue background. At the top right, there are 'Login' and 'Sign Up' buttons. The sign-up section is titled 'Sign Up' in blue, followed by the subtitle 'Discover Your Next Adventure With Us'. It contains several form fields: 'Name:', 'User Name:', 'Email:', 'Phone Number:', 'Gender:' (with radio buttons for 'Male' and 'Female'), 'Password:', 'Confirm Password:', and 'Address:'. Each field has a small 'Type' label on the right. At the bottom of the form, there are 'Cancel' and 'Sign up' buttons. A vertical image of an airplane on a tarmac is positioned to the left of the form fields.

**Figure 2: Signup Page****Figure 3: Home Page****Figure 4: Profile**

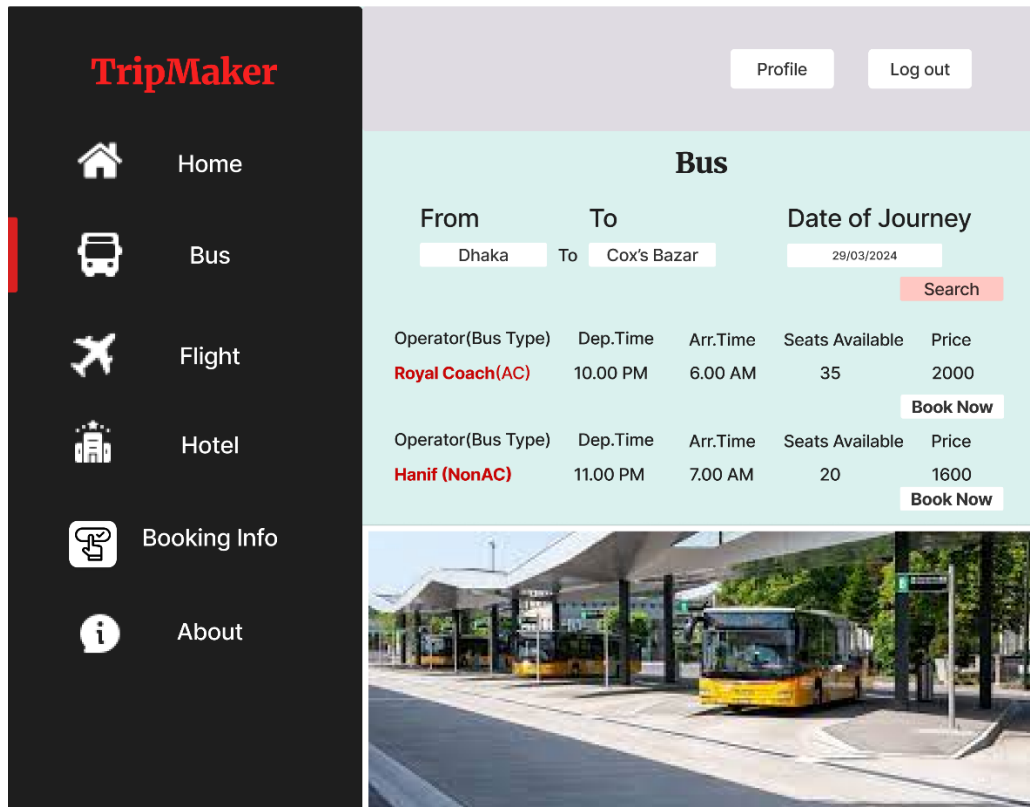


Figure 5: Bus

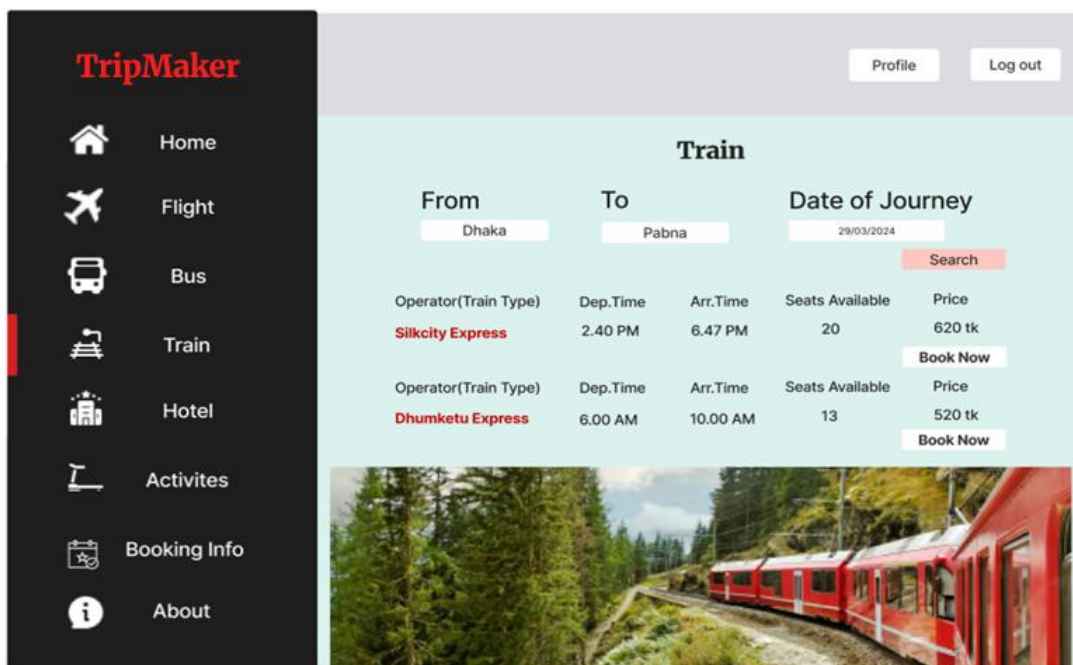


Figure 6: Train

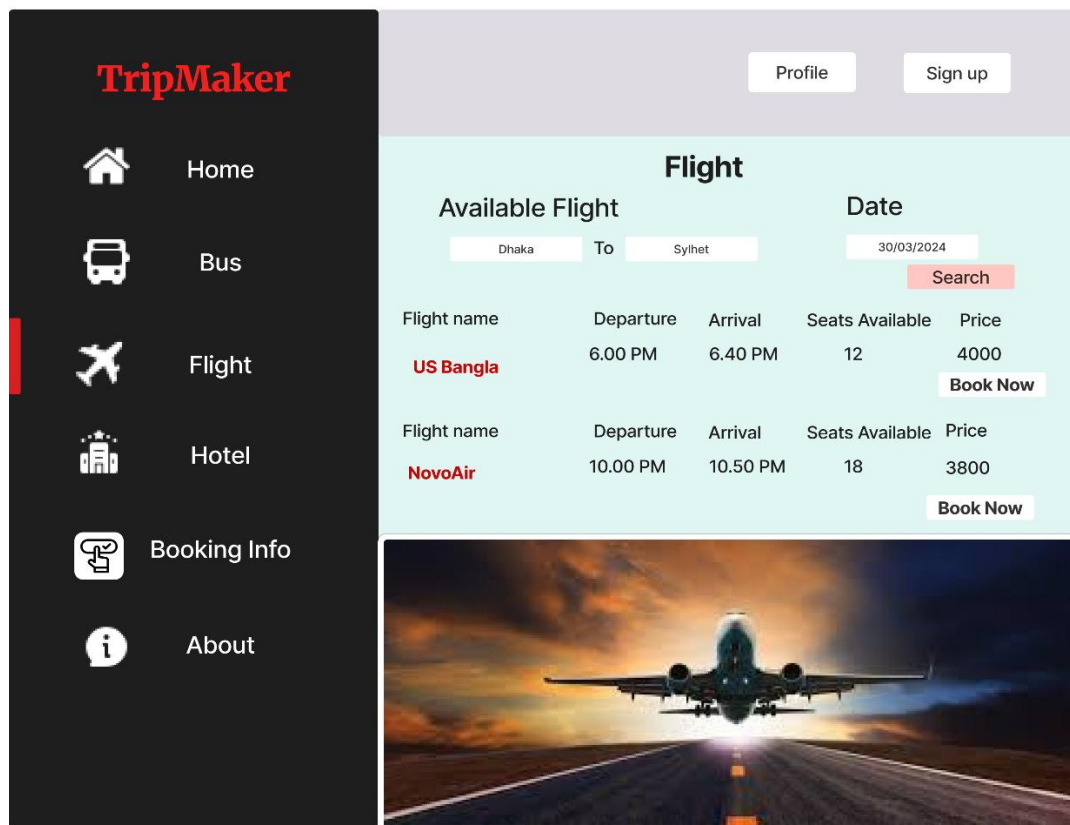


Figure 7: Flight

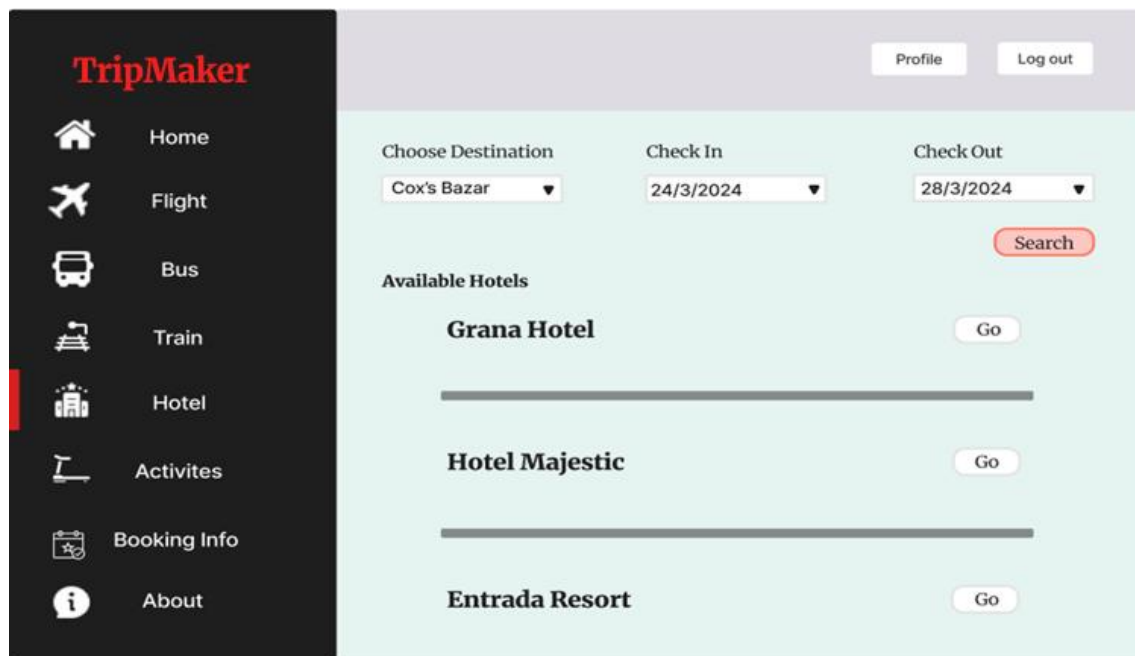


Figure 8: Hotel

**TripMaker**

- Home
- Flight
- Bus
- Train
- Hotel**
- Activites
- Booking Info
- About

Profile Log out

**Available Rooms Grana Hotel** Book Now Back

**Super Delux**

<input type="checkbox"/> Breakfast Included	Lunch Included	Dinner Included	Guest	
Yes	No	Yes	1	<b>100TK</b>

---

**Normal**

<input type="checkbox"/> Breakfast Included	Lunch Included	Dinner Included	Guest	
No	No	No	5	<b>500TK</b>

---

**Delux**

<input type="checkbox"/> Breakfast Included	Lunch Included	Dinner Included	Guest	
Yes	No	No	4	<b>500TK</b>

Figure 9: Hotel Booking

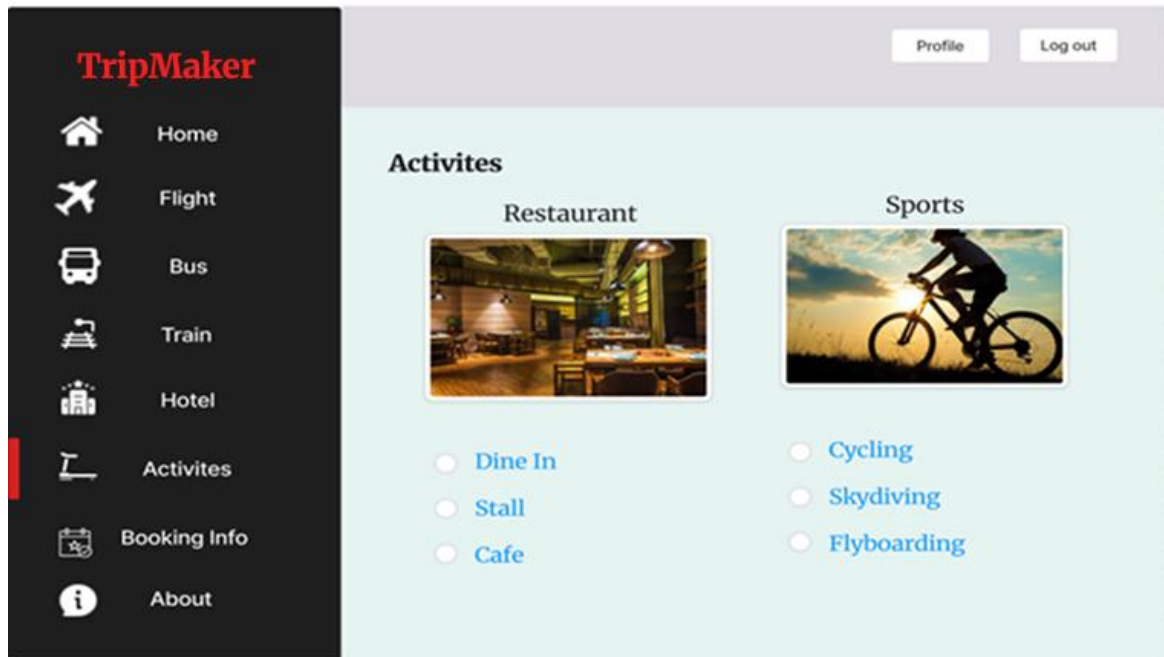


Figure 10: Activites

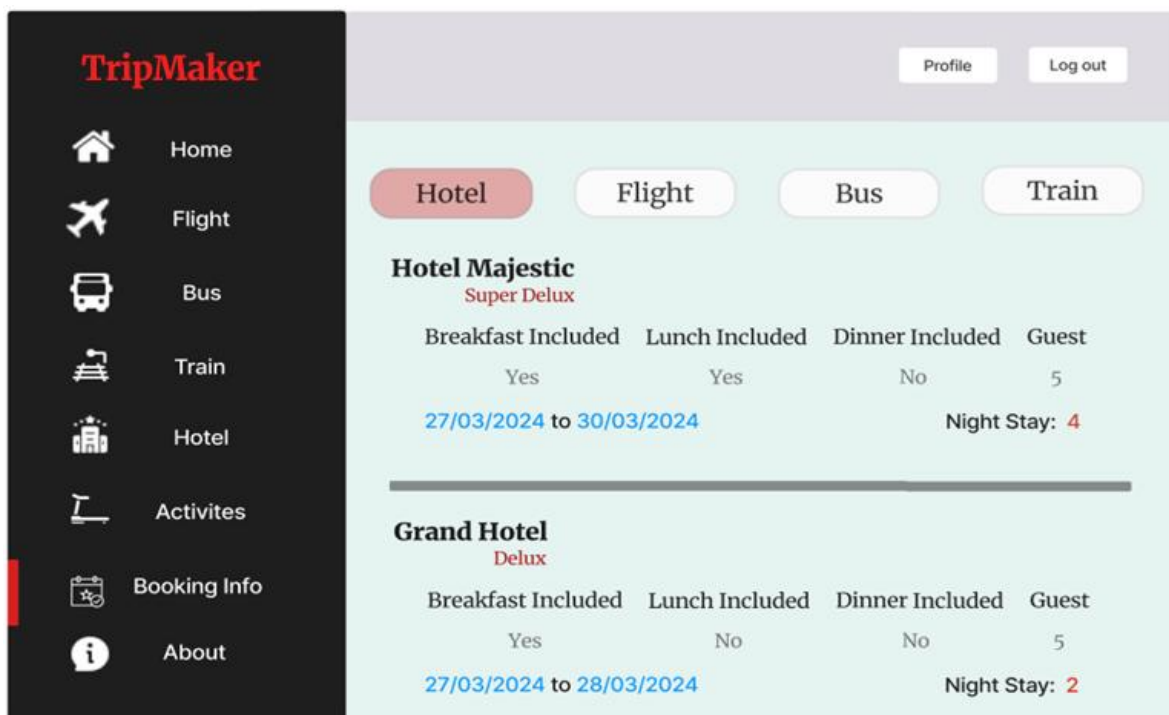
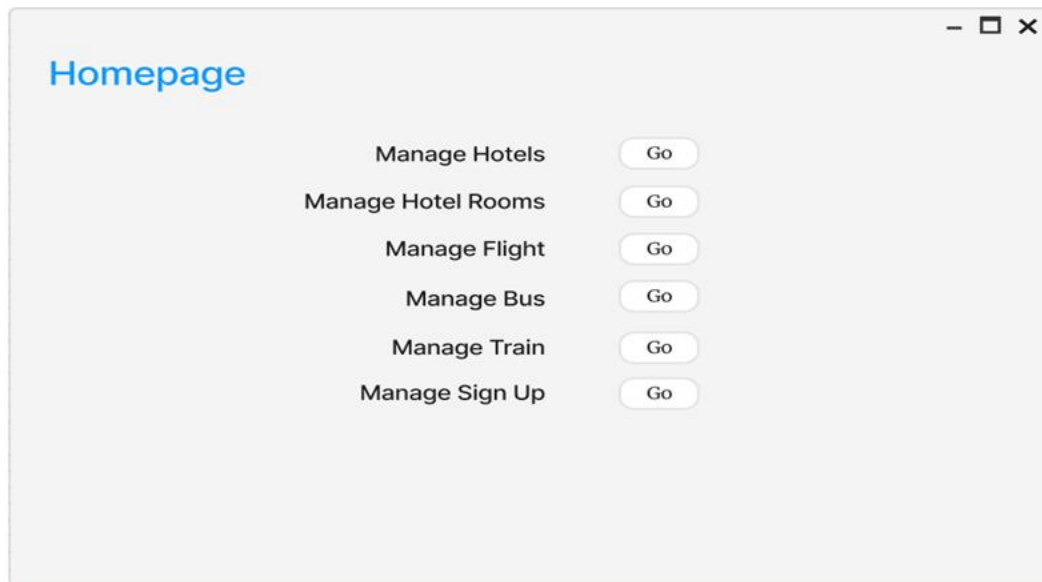
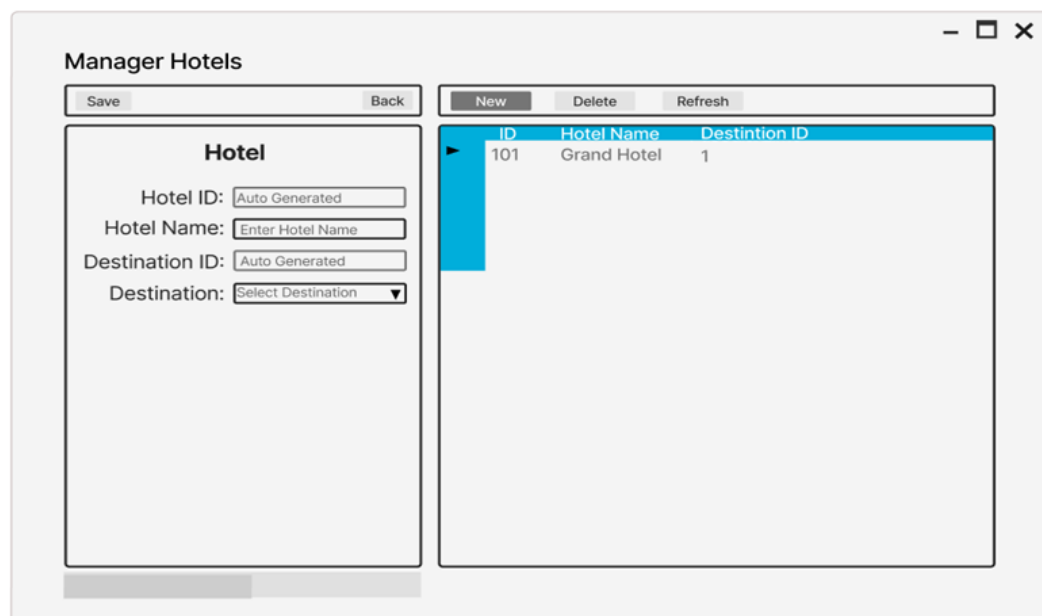


Figure 11: Booking Info



**Figure 12:** Admin Home Page



**Figure 13:** Manage Hotel



**Manage Hotel Room**

Save Back

Room ID: Auto Generated

Cost Per Night : Enter cost per night

Breakfast Include: ☐ Yes ☐ No

Lunch Include: ☐ Yes ☐ No

Dinner Include: ☐ Yes ☐ No

Hotel ID : Select hotel ID ▼

Hotel : Select hotel ▼

Room Type : Enter Room type

Guest per room : Enter guest per room

New Delete Refresh

ID	Cost Per Night	Breakfast	Lunch
1	1800	Yes	Yes
2	1000	No	No
3	2000	Yes	Yes
4	1500	No	Yes

Figure 14: Manage Hotel Room

**Manager Flights**

Save Back

**Flight**

Flight ID: Auto Generated

Airlines: xyz always

Departure ID: Auto Generated

Arrival ID: Auto Generated

Departed Time: Enter Time

Arrived Time: Enter Time

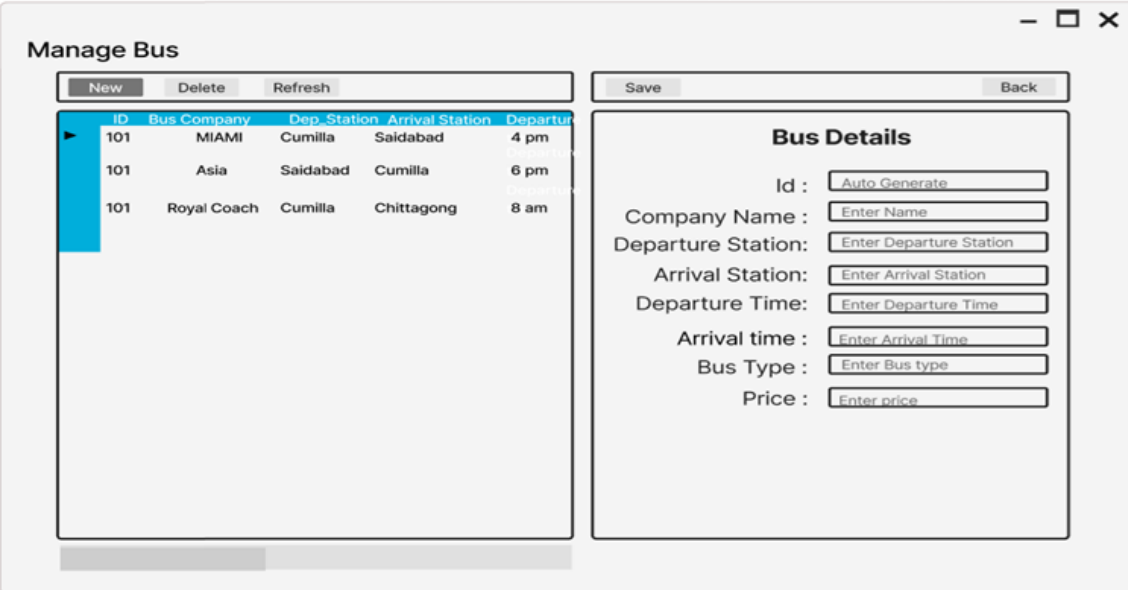
Price: \$

Date: 27/3/2024 ▼

New Delete Refresh

ID	Departure ID	Arrival ID	Date	Time
101	1	2	27/3/24	6.00 P.

Figure 15: Manage Flight



The 'Manage Bus' interface is a web application window with a title bar containing standard window controls. It is divided into two main sections. The left section features a table with columns: ID, Bus Company, Dep. Station, Arrival Station, and Departure Time. Above the table are three buttons: 'New', 'Delete', and 'Refresh'. The table contains three rows of data. The first row is highlighted with a blue background. The right section is titled 'Bus Details' and contains a 'Save' button at the top left and a 'Back' button at the top right. Below these are seven input fields, each with a label and a text input box: 'Id' (with an 'Auto Generate' button), 'Company Name', 'Departure Station', 'Arrival Station', 'Departure Time', 'Arrival time', 'Bus Type', and 'Price'.

ID	Bus Company	Dep. Station	Arrival Station	Departure Time
101	MIAMI	Cumilla	Saidabad	4 pm
101	Asia	Saidabad	Cumilla	6 pm
101	Royal Coach	Cumilla	Chittagong	8 am

**Bus Details**

Id :

Company Name :

Departure Station:

Arrival Station:

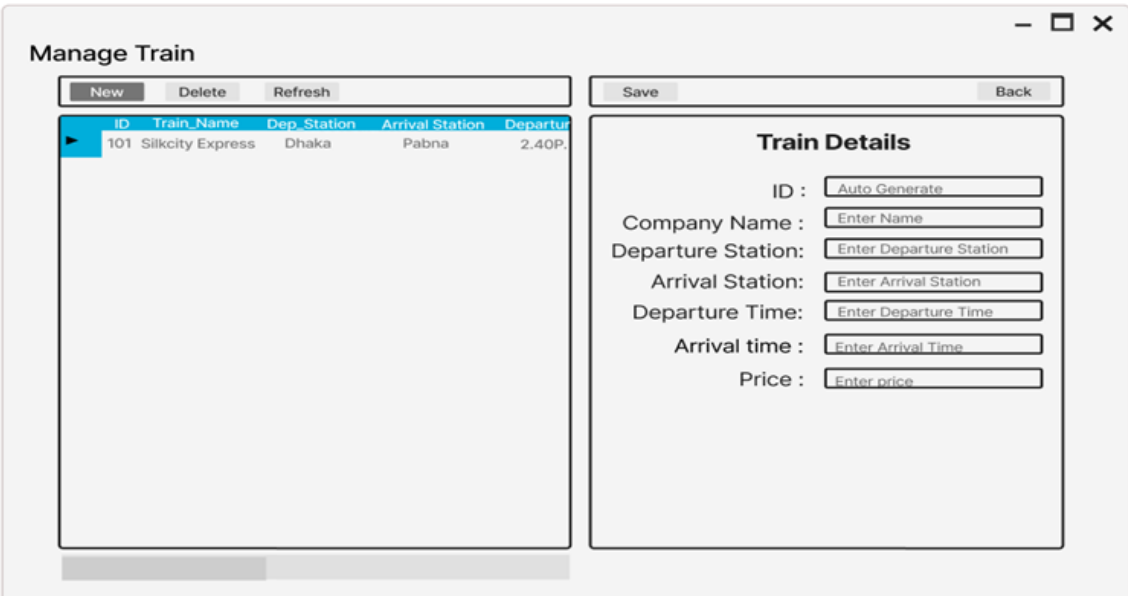
Departure Time:

Arrival time :

Bus Type :

Price :

Figure 16: Manage Bus



The 'Manage Train' interface is a web application window with a title bar containing standard window controls. It is divided into two main sections. The left section features a table with columns: ID, Train Name, Dep. Station, Arrival Station, and Departure Time. Above the table are three buttons: 'New', 'Delete', and 'Refresh'. The table contains one row of data. The right section is titled 'Train Details' and contains a 'Save' button at the top left and a 'Back' button at the top right. Below these are seven input fields, each with a label and a text input box: 'ID' (with an 'Auto Generate' button), 'Company Name', 'Departure Station', 'Arrival Station', 'Departure Time', 'Arrival time', and 'Price'.

ID	Train Name	Dep. Station	Arrival Station	Departure Time
101	Silkcity Express	Dhaka	Pabna	2.40P.

**Train Details**

ID :

Company Name :

Departure Station:

Arrival Station:

Departure Time:

Arrival time :

Price :

Figure 17: Manage Train

New

Delete

Refresh

Save

Back

ID	User Name	Password	Email	Type
101	abc	123	abc@gmail.c..	admi..

### Sign Up

Id :

Name :

User Name :

Email :

Phone Number :

Gender : ☐ Male ☐ Female

Password :

Confirm Password :

Address :

Figure 18: Manage User

#### 4.4 Project Requirements

##### Resources Required Table:

Phase	Duration(weeks)	Resources Required
Planning	2 weeks	<ul style="list-style-type: none"> <li>• Business Analyst: (1)</li> <li>• Project manager: (1)</li> </ul>
Analysis	2 weeks	<ul style="list-style-type: none"> <li>• Business Analyst: (1)</li> <li>• Project manager: (1)</li> </ul>

<b>Design</b>	3 weeks	<ul style="list-style-type: none"> <li>• System Architect (1)</li> <li>• UI/UX Designer (1)</li> <li>• Database Architect (1)</li> </ul>
<b>Implementation</b>	4 weeks	<ul style="list-style-type: none"> <li>• Frontend Developers (1)</li> <li>• Backend Developers (2)</li> <li>• Database Administrator (1)</li> </ul>
<b>Testing</b>	3 weeks	<ul style="list-style-type: none"> <li>• Quality Assurance Engineers (2)</li> </ul>
<b>Deployment</b>	1 weeks	<ul style="list-style-type: none"> <li>• System Administrator (1)</li> <li>• Backend Developers (for support) (1)</li> <li>• Database Administrator (1)</li> </ul>
<b>Maintenance</b>	3 weeks	<ul style="list-style-type: none"> <li>• System Administrator (1)</li> <li>• QA Engineer (1)</li> <li>• Backend Developer (1)</li> <li>• Database Administrator (1)</li> </ul>

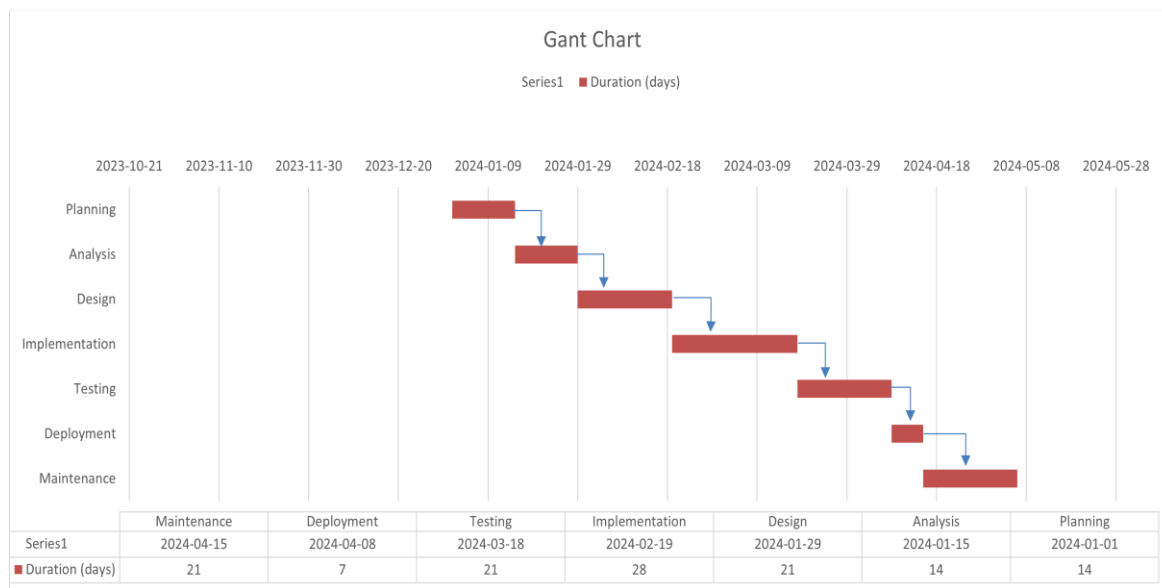
- **Language and design:** HTML, CSS, PHP, JavaScript, MySQL.
- **Software:** Visual Studio Code, Xampp, GitHub.

### Scheduling Table:

Task Serial	Task Name	Duration (days)	Start	Finish	Preprocessor
1	Planning	2 weeks	Jan 1, 2025	Jan 14, 2025	
2	Analysis	2 weeks	Jan 15, 2025	Jan 28, 2025	Planning
3	Design	3 weeks	Jan 29, 2025	Feb 18, 2025	Analysis

4	Implementation	4 weeks	Feb 19, 2025	Mar 17, 2025	Design
5	Testing	3 weeks	Mar 18,2025	Apr 7,2025	Implementation
6	Deployment	1 weeks	Apr 8, 2025	Apr 14, 2025	Testing
7	Maintenance	3 weeks	Apr 15,2025	May 5,2025	Deployment

**Total Development Time: 18 weeks.**



### Budget Estimation:

Phase	Activities	Budget Items	Estimated Cost (BDT)
<b>Planning</b>	<ul style="list-style-type: none"> <li>Project scope definition</li> <li>Feasibility studies</li> <li>Resource allocation</li> <li>Project scheduling</li> </ul>	<ul style="list-style-type: none"> <li>Business Analyst: <math>50,000 \times 0.46 = 23,000</math></li> <li>Project Manager: <math>65,000 \times 0.46 = 29,900</math></li> <li>Cost of feasibility studies or consultants: 22,000 BDT</li> <li>Expenses for stakeholder meetings: 12,000 BDT</li> <li>Software and tools for project planning: 6,000 BDT</li> </ul>	<ul style="list-style-type: none"> <li>Estimated Cost: 92,900 BDT</li> </ul>

<b>Analysis</b>	<ul style="list-style-type: none"> <li>Gathering requirements</li> <li>Documenting requirements</li> <li>Creating use cases</li> </ul>	<ul style="list-style-type: none"> <li>Business Analyst: <math>50,000 \times 0.46 = 23,000</math></li> <li>Project Manager: <math>65,000 \times 0.46 = 29,900</math></li> <li>Tools for requirements management: 9,000 BDT</li> </ul>	<ul style="list-style-type: none"> <li><b>Estimated Cost:</b> 61,900 BDT</li> </ul>
<b>Design</b>	<ul style="list-style-type: none"> <li>System architecture design</li> <li>User interface design</li> <li>Database design</li> </ul>	<ul style="list-style-type: none"> <li>System Architect: <math>100,000 \times 0.69 = 69,000</math></li> <li>UI/UX Designer: <math>55,000 \times 0.69 = 37,950</math></li> <li>Database Architect: <math>95,000 \times 0.69 = 65,550</math></li> <li>Software and tools for design (e.g., modeling software): 7,000 BDT</li> <li>Cost of design reviews or expert consultations: 20,000 BDT</li> </ul>	<ul style="list-style-type: none"> <li><b>Estimated Cost:</b> 199,500 BDT</li> </ul>
<b>Implementation</b>	<ul style="list-style-type: none"> <li>Coding</li> <li>Software development</li> <li>Integration</li> </ul>	<ul style="list-style-type: none"> <li>Frontend Developer: <math>60,000 \times 0.92 = 55,200</math></li> <li>Backend Developer (2): <math>65,000 \times 2 \times 0.92 = 119,600</math></li> <li>Database Administrator: <math>75,000 \times 0.92 = 69,000</math></li> <li>Cost of programming tools, IDEs: 7,000 BDT</li> <li>Server and infrastructure costs: 45,000 BDT</li> <li>Version control and collaboration software: 6,000 BDT</li> </ul>	<ul style="list-style-type: none"> <li><b>Estimated Cost:</b> 301,800 BDT</li> </ul>
<b>Testing</b>	<ul style="list-style-type: none"> <li>Unit testing</li> <li>Integration testing</li> <li>System testing</li> <li>User acceptance testing</li> </ul>	<ul style="list-style-type: none"> <li>QA Engineer (2): <math>45,000 \times 2 \times 0.69 = 62,100</math></li> <li>Test automation tools: 7,000 BDT</li> <li>Lab or testing environment costs: 14,000 BDT</li> </ul>	<ul style="list-style-type: none"> <li><b>Estimated Cost:</b> 83,100 BDT</li> </ul>

<b>Deployment</b>	<ul style="list-style-type: none"> <li>• Software installation</li> <li>• Configuration</li> <li>• User training</li> </ul>	<ul style="list-style-type: none"> <li>• System Administrator: <math>55,000 \times 0.23 = 12,650</math></li> <li>• Backend Developer: <math>65,000 \times 0.23 = 14,950</math></li> <li>• Database Administrator: <math>75,000 \times 0.23 = 17,250</math></li> <li>• Deployment tools and infrastructure: 13,000 BDT</li> <li>• Cost of training materials and trainers: 14,000 BDT</li> <li>• Travel expenses for on-site deployment: 10,000 BDT</li> </ul>	<ul style="list-style-type: none"> <li>• <b>Estimated Cost:</b> 81,850 BDT</li> </ul>
<b>Maintenance</b>	<ul style="list-style-type: none"> <li>• Bug fixing</li> <li>• Software updates</li> <li>• Technical support</li> </ul>	<ul style="list-style-type: none"> <li>• System Administrator: <math>55,000 \times 0.69 = 37,950</math></li> <li>• QA Engineer: <math>45,000 \times 0.69 = 31,050</math></li> <li>• Backend Developer: <math>65,000 \times 0.69 = 44,850</math></li> <li>• Database Administrator: <math>75,000 \times 0.69 = 51,750</math></li> <li>• Server and infrastructure maintenance: 15,000 BDT</li> <li>• Cost of software updates and patches: 13,000 BDT</li> </ul>	<ul style="list-style-type: none"> <li>• <b>Estimated Cost:</b> 193,600 BDT</li> </ul>

The total estimated cost for this table is: **1,014,650 BDT -1,50,000 BDT**

### COCOMO (CONSTRUCTIVE COST MODEL):

Software Project Type	Coefficient <Effort Factor>	P	T
Organic	2.4	1.05	0.38
Semi-Detached	3.0	1.12	0.35
Embedded	3.6	1.20	0.32

**Effort Estimation:**  $a \times (\frac{SLOC}{1000})^b$

**As the project is organic,**  $a = 2.4$ ,  $b = 1.05$

Here,  $SLOC = 20,000$

$$\begin{aligned}\text{Effort} &= a \times (\frac{SLOC}{1000})^b \\ &= 2.4 \times (20)^{1.05} \\ &\approx 4.96 \text{ person-months} \\ &\approx 5 \text{ person-months}\end{aligned}$$

**Development Time:**  $c \times (Effort)^d$

Here,  $c = 2.5$ ,  $d = 0.38$  [**organic**]

$$\begin{aligned}\text{Development Time} &= 2.5 \times (5)^{0.38} \\ &= 2.5 \times 1.84 \\ &\approx 4.6 \text{ months}\end{aligned}$$

**Number of People Required:**  $\frac{Effort}{Development\ Time}$

$$\begin{aligned}\text{Number of People} &= \frac{4.96}{4.594} \\ &\approx 1.08 \text{ people} \\ &\approx 2 \text{ people}\end{aligned}$$

## 5. FEATURES NOT TO BE TESTED

The following features and areas of the TripMaker system are outside the scope of direct testing in this project. Any testing in these areas will be considered indirect or will fall under the responsibility of the client or third-party vendors:



**List of Features Not to Be Tested:****1. Third-Party Payment Gateway Integration**

- The connection to external payment processors (e.g., bKash, Nagad, etc.) will not be tested in detail. Only the success/failure response will be verified from TripMaker's end.

**2. Email Delivery and SMS Notification Systems**

- Verification of email/SMS delivery delays, formatting across providers, and spam filtering will not be tested. We assume the configured services are functional under the client's control.

**3. Cross-System Interoperability**

- Interfaces with external systems (e.g., third-party transport or hotel booking APIs) will not be tested unless directly affecting TripMaker's own database or response logic.

**4. Mobile Responsiveness on All Devices**

- TripMaker will be tested on common browsers and mobile screens, but extensive testing on all screen sizes and devices (e.g., tablets, foldables, smart TVs) is beyond current scope.

**6. TESTING APPROACH****6.1 Testing Levels**

Testing TripMaker involves multiple structured levels to ensure that each part of the system is functional, stable, and aligns with user expectations. The testing process will follow a layered approach, starting from unit-level validation to full-scale acceptance testing.

**Unit Testing**

This is the most granular level of testing, where individual modules or components of the TripMaker system such as customer registration, login, transport booking, and profile updates will be tested in isolation. The objective is to validate the correctness of functions, logic, and data processing within each module.

- **Approach:** White-box testing techniques will be used, including control flow, data flow, and decision path testing. Test drivers and stubs will be implemented as needed to simulate dependencies.
- **Responsibility:** Primarily handled by developers during the coding phase.
- **Scope:** Validation logic, calculations (ticket price, seat availability), and database operations (inserts/updates), Error handling.

- **Tools:** JUnit (for Java-based modules), NUnit (.NET), built-in IDE debuggers, and Postman (for API endpoints).

## Integration Testing

This level focuses on testing how different modules of the system interact with each other when integrated. For TripMaker, this involves testing communication between modules like transport booking, hotel reservation, activity selection, user profiles, and admin services.

- **Approach:** Incremental integration testing will be adopted, along with a hybrid of **top-down** strategies.
- **Scope:** Interface interactions between customer and booking modules, booking and Transport/Hotel/Activity modules and database, admin and service update flows.
- **Special Focus:** Interface compatibility, message/data passing (especially in service booking), and shared resources.
- **Tools:** Postman for API interaction testing, SQLite DB Browser for database-level verification, and SoapUI for complex service integrations.

## System Testing

System testing evaluates the TripMaker system as a whole to ensure that all modules work together seamlessly in a fully integrated environment. This phase simulates actual user behavior to validate both functional and non-functional requirements.

- **Approach:** A dedicated QA team will simulate real user scenarios and workflows to assess the system's behavior from an external perspective.
- **Scope:** End-to-end workflows such as user registration, login/logout, searching and booking services (transportation, hotels, activities), viewing booking history, updating user profiles, and admin functionalities.
- **Test Categories:**
  - **Functional Testing** to ensure the correctness of operations based on requirements
  - **Usability Testing** to evaluate ease of use, layout consistency, and intuitive navigation
  - **Performance Testing** to assess response time and resource utilization under load
  - **Security Testing** to validate that unauthorized access is prevented and sensitive data is protected
  - **Compatibility Testing** across various browsers, screen sizes, and devices
- **Tools:** Selenium for automation, JMeter for performance/load testing, OWASP ZAP for vulnerability scanning.

## Regression Testing

Regression Testing will be performed whenever a new feature is added, a defect is fixed, or an update is made to ensure that previously developed and tested features still work correctly.

- **Purpose:** To ensure that nothing is broken or unintentionally affected by changes in the codebase.
- **Scope:** Re-run of previously executed test cases, focusing on critical workflows such as user authentication, booking, payment processing, and admin control panel.
- **Tools:** Selenium (for automation), version-controlled test suites in Postman or JUnit/NUnit frameworks.

## Acceptance Testing

Acceptance Testing is the final phase of testing and is performed to determine whether the system meets the business needs and expectations of the customer. The software must be in a stable and mostly defect-free state to proceed with this phase.

- **Types:**
  - Alpha Testing (internal team simulates real user workflows)
  - Beta Testing (external or pilot users test the system in real environments)
- **Criteria:** System usability, complete booking process flow, data accuracy, and admin-side control.
- **Execution Phases:**
  - Phase 1: Basic features such as login, profile management, and single service booking
  - Phase 2: Complex features like multi-service bookings, cancellation, and viewing historical data
- **Tools:** Manual testing with test reports maintained in Google Sheets; feedback via forms or direct reports.

## 6.2 Test Tools

Below is a list of tools selected for use during different phases of the TripMaker project to support design, development, testing, and management:

Category	Tool(s)	Purpose
UI/UX Design	Figma	Designing wireframes and user flows
Code Development	Visual Studio, IntelliJ, VS Code	Application development in C#, Java, HTML/CSS, JavaScript
Database Management	SQLite, Oracle SQL Developer	Schema creation, test data handling

Unit Testing	JUnit, NUnit, Postman	Individual module testing and API request validation
Integration Testing	Postman, SoapUI	API endpoint testing and system interface verification
System Testing	Selenium, JMeter, OWASP ZAP	UI automation, load and performance testing, and security checks
Regression Testing	Selenium (scripts), GitHub Actions	Retesting existing features after updates
Bug Tracking	GitHub Issues, Trello	Logging, tracking, and resolving bugs
Documentation	Google Docs, Notion	Test plan, strategy, and result documentation
Reporting	Google Sheets, Excel	Test result analysis and status tracking
Project Management	Trello, Notion	Sprint planning, team coordination, and milestone tracking

### 6.3 Meetings

Effective communication is essential for a smooth testing lifecycle. The TripMaker project team will conduct meetings at regular intervals to ensure progress tracking, defect review, coordination, and strategic planning.

Frequency	Meeting Type	Attendees	Purpose
Daily	Stand-up Meeting	Developers, Testers	Share daily progress, report blockers, assign testing goals
Weekly	QA Progress Review	QA Engineers, Dev Leads, PM	Review test execution, defect status, test coverage, upcoming priorities
Bi-Weekly	Sprint Demo & Review	Full Project Team, Stakeholders	Demonstrate implemented features, gather feedback, discuss user requirements

## 7. TEST CASES/TEST ITEMS

Table 1: Test Case for **Login Session**

Project Name: Tripmaker: A Travel Agency Management System		Test Designed by: Sha Mohammad Yeahia Idris		
Test Case ID: TPM-LOGIN-TC01		Test Designed date: 28/4/2025		
Test Priority (Low, Medium, High): High		Test Executed by:		
Module Name: Login Session		Test Execution date:		
Test Title: verify login with valid username and password				
Description: Test application login page				
Precondition (If any): User must have valid username and password				
Test Steps	Test Data	Expected Results	Actual Results	Status (Pass/Fail)
1. Go to the application 2. Enter username 3. Enter password 4. Click Sign in	Username: idris11 Password: @13475090i5	User should login into the application, username and password have been matched		
	Username: idris11 Password: @13475	User will not be able to login into the application because username and password did not match		
Post Condition: User is validated with database and successfully login to account. The account session details are logged in the database.				

Table 2: Test Case for **Sign up Session**

Project Name: Tripmaker: A Travel Agency Management System		Test Designed by: Sha Mohammad Yeahia Idris		
Test Case ID: TPM- SIGN_UP -TC02		Test Designed date: 28/4/25		
Test Priority (Low, Medium, High): High		Test Executed by:		
Module Name: Sign up Session		Test Execution date:		
Test Title: Verify signup information				
Description: Test application Sign up page				
Precondition (If any):				
Test Steps	Test Data	Expected Results	Actual Results	Status (Pass/Fail)
1. Go to the application 2. Go to Sign up page 3. Enter sign up information 4. Click Signup	Name: Idris Username: Idris98 Email: <a href="mailto:idris32@gmail.com">idris32@gmail.com</a> Number: 017123456 Gender: Male Password: abc@12 C Password: abc@12 Address: Kuril, Dhaka.	User successfully create an account.		
	Name: Idris Username: Idris98 Email: <a href="mailto:idris32gmail.com">idris32gmail.com</a> Number: 017123456 Gender: Male Password: abc@12 C Password: abc@1 Address: Kuril, Dhaka.	User will not be able to create account. Invalid email address. Password and confirm password are not match.		
Post Condition: User successfully create an new account. Sign up information are store in database.				

Table 3: Test Case for **Update Session**

Project Name: Tripmaker: A Travel Agency Management System		Test Designed by: Sha Mohammad Yeahia Idris		
Test Case ID: TPM-UPDATE-TC03		Test Designed date: 30/4/25		
Test Priority (Low, Medium, High): Medium		Test Executed by:		
Module Name: Update Session		Test Execution date:		
Test Title: Access to a profile page displaying their personal information				
Description: Test application update page				
Precondition (If any): User must have login first				
Test Steps	Test Data	Expected Results	Actual Result	Status (Pass/Fail)
1. Log in 2. Go to profile 3. Edit profile details 4. Click Update Profile	Change Phone Number: 01712345678 Change Email: idris01@gmail.com Password: Abc@1264	User successfully updated information		
	Change Phone Number: 018123456aa Change Email: idris01gmail.com Password: Abc@1264	User will not able to updated the information because user inserted invalid phone number and email		
Post Condition: Change to the profile will update the database and user interface.				

Table 4: Test Case for **Logout Session**

Project Name: Tripmaker: A Travel Agency Management System		Test Designed by: Sha Mohammad Yeahia Idris	
Test Case ID: TPM-LOGINOUT-TC04		Test Designed date: 1/5/25	
Test Priority (Low, Medium, High): Low		Test Executed by:	
Module Name: Logout Session		Test Execution date:	
Test Title: Verify logout confirmation			
Description: Test application logout button			
Precondition (If any): User needs to login first			

Test Steps	Test Data	Expected Results	Actual Result	Status(Pass/Fail)
1. Log in 2. Click Logout button	Username: Idris98 Password: @13475090i5	User successfully has been logged out.		
Post Condition: The account session details are logged out from the database. Change to user interface and the functionalities will be limited.				

Table 5: Test Case for Homepage Session

Project Name: Tripmaker: A Travel Agency Management System		Test Designed by: Dip Achorjee Sokal		
Test Case ID: TPM-HOMEPAGE-TC05		Test Designed date: 1/5/25		
Test Priority (Low, Medium, High): High		Test Executed by:		
Module Name: Homepage Session		Test Execution date:		
Test Title: Verify navigation of Homepage				
Description: Test application Homepage				
Precondition (If any): User has the access to the homepage				
Test Steps	Test Data	Expected Results	Actual Result	Status(Pass/Fail)
1. Open application		Homepage appeared to the user		
Post Condition: User can navigate to the pages related to booking and can view information.				

Table 6: Test Case for Search Session in Transportation Pages

Project Name: Tripmaker: A Travel Agency Management System	Test Designed by: Dip Achorjee Sokal
Test Case ID: TPM- TRANSPORT -TC06	Test Designed date: 2/5/25
Test Priority (Low, Medium, High): Medium	Test Executed by:
Module Name: Search Session in Transportation Pages (Flight, Bus, Train)	Test Execution date:
Test Title: verify search in transportation pages (Flight, Bus, Train)	
Description: Test application Transportation Pages (Flight, Bus, Train)	
Precondition (If any): User has the access to the homepage	



Test Steps	Test Data	Expected Results	Actual Result	Status (Pass/Fail)
1. Open application to view homepage 2. Select a specific transportation option through sidebar. 3. Select information available in combo box 4. Click search button	From: Dhaka To: Shylet Date: 20/04/2025	Show available transportation.		
	From: Dhaka To: Date: 20/04/2025	Show error message.		
Post Condition: Details of available transportations will be selected from the database and will be visible in user interface.				

Table 7: Test Case for **Search Session in Hotel page**

Project Name: Tripmaker: A Travel Agency Management System		Test Designed by: Dip Achorjee Sokal		
Test Case ID: TPM-SEARCH_HOTEL-TC07		Test Designed date: 2/5/25		
Test Priority (Low, Medium, High): Medium		Test Executed by:		
Module Name: Search Session in Hotel page		Test Execution date:		
Test Title: Verify Search in Hotel Page				
Description: Test application Hotel Page				
Precondition (If any): User has the access to the homepage				
Test Steps	Test Data	Expected Results	Actual Result	Status(Pass/Fail)
1. Open application to view homepage	Choose Destination: Cox's Bazar Check In: 05/05/2025 Check Out: 08/05/2025	Show available hotels.		

2. Select hotel option through sidebar 3. Select information available in combo box 4. Click search button	Choose Destination: Cox's Bazar Check In: 05/05/2025 Check Out: 04/05/2025	Show error message.		
Post Condition: List of available hotel will be selected from the database and will be visible in user interface.				

Table 8: Test Case for **View Details Session of Hotel**

Project Name: Tripmaker: A Travel Agency Management System		Test Designed by: Dip Achorjee Sokal		
Test Case ID: TPM-HOTEL-TC08		Test Designed date: 2/5/25		
Test Priority (Low, Medium, High): Medium		Test Executed by:		
Module Name: View Details Session of Hotel		Test Execution date:		
Test Title: Verify Go button in Hotel page				
Description: Test application Hotel page				
Precondition (If any): User has the access to the homepage				
Test Steps	Test Data	Expected Results	Actual Result	Status(Pass/Fail)
1. Open application to view homepage 2. Select hotel option through sidebar 3. Select information available in combo box 4. Click Search button 5. Click Go button	Choose Destination: Cox’s Bazar Check In: 10/05/2025 Check Out: 13/05/2025	Show details of a hotel.		
Post Condition: Details of a available hotel will be selected from the database will be visible in user interface.				

Table 9: Test Case for **Activity Selecting Session**

Project Name: Tripmaker: A Travel Agency Management System		Test Designed by: Md. Rezwan Mujahid Rudro		
Test Case ID: TPM-ACTIVITY-TC09		Test Designed date: 7/5/25		
Test Priority (Low, Medium, High): Low		Test Executed by:		
Module Name: Activity Selecting Session		Test Execution date:		
Test Title: Verify activity selection				
Description: Test application Activity selection				
Precondition (If any): User has the access to the homepage				
Test Steps	Test Data	Expected Results	Actual Result	Status(Pass/Fail)
1. Open application to view Homepage 2. Select Activities option through Sidebar. 3. Select information available in combo box	Location: Cox’s Bazar Date: 27/04/2025 Category: Sports	Show available activites.		
Post Condition: Details of available activity will be selected from the database and will be visible in user interface.				

Table 10: Test Case for **Book now Session**

Project Name: Tripmaker: A Travel Agency Management System		Test Designed by: Rezwan Mujahid Rudro	
Test Case ID: TPM-BOOKING-TC10		Test Designed date: 7/5/25	
Test Priority (Low, Medium, High): High		Test Executed by:	
Module Name: Book now Session		Test Execution date:	
Test Title: Verify booking in Transportation pages (flight, bus, train) and Hotel pages			
Description: Test application of Book Now button			
Precondition (If any): User needs to login first			

Test Steps	Test Data	Expected Results	Actual Result	Status(Pass/Fail)
1. Open application to view Homepage 2. Log in 3. Select a specific option through Sidebar. 4. Select information available in combo box 5. Click Search button 6. Click Book Now button	Choose Destination: Cox's Bazar Check In: 17/04/2025 Check Out: 20/04/2025	Booking Info page appeared to the user		
Post Condition: Store booking information in database.				

Table 11: Test Case for **View Booking Info Session**

Project Name: Tripmaker: A Travel Agency Management System		Test Designed by: Md. Rezwan Mujahid Rudro		
Test Case ID: TPM-BOOKING_INFO-TC11		Test Designed date: 8/5/25		
Test Priority (Low, Medium, High): High		Test Executed by:		
Module Name: View Booking Info Session		Test Execution date:		
Test Title: Verify accessibility of Booking Info page				
Description: Test application Booking Info page				
Precondition (If any): User needs to login first				
Test Steps	Test Data	Expected Results	Actual Result	Status(Pass/Fail)
1. Open application to view homepage 2. Log in 3. Select booking info		Booking details history will be showed.		

option through sidebar				
Post Condition: Extract booking information from database.				

Table 12: Test Case for **View Wallet Session**

Project Name: Tripmaker: A Travel Agency Management System		Test Designed by: S M Abid Hassan		
Test Case ID: TPM-WALLET-TC12		Test Designed date: 30/4/25		
Test Priority (Low, Medium, High): High		Test Executed by:		
Module Name: View Wallet Session		Test Execution date:		
Test Title: Verify accessibility of Wallet				
Description: Test application of Wallet				
Precondition (If any): User needs to login first				
Test Steps	Test Data	Expected Results	Actual Result	Status(Pass/Fail)
1.Open application to view Homepage 2. Log in 3. Select a specific option through Sidebar. 4. Select information available in combo box 5. Click Search button 6. Click Book Now button 7. Click Wallet button	Payment Method: Cash	Show payment successful message.		
	Payment Method: Online	System will ask for credit card information		
Post Condition: Booking will be successful and store all the information into database.				

Table 13: Test Case for **Insert and Update Session in Manager View**

Project Name: Tripmaker: A Travel Agency Management System		Test Designed by: S M Abid Hassan		
Test Case ID: TPM-MANAGER_VIEW-TC13		Test Designed date: 3/5/25		
Test Priority (Low, Medium, High): High		Test Executed by:		
Module Name: Insert and Update Session in Manager View		Test Execution date:		
Test Title: verify manager view functionalities (insert and update)				
Description: Test application Manager View pages				
Precondition (If any): Need to Log in as admin				
Test Steps	Test Data	Expected Results	Actual Result	Status(Pass/Fail)
1. Open application 2. Log in as admin to view Homepage 3. Click Go button beside a specific manager view 4. Insert or update data 5. Click Save button	Hotel ID: Auto Generated Hotel Name: Grana Hotel Destination ID: Auto generated Destination: Cox’s Bazar	Data inserted successfully.		
	Hotel ID: Auto Generated Hotel Name: Grana Hotel Destination ID: Auto generated Destination:	Error message will be shown.		
Post Condition: Inserted or updated data will be stored in the database .				

Table 14: Test Case for **Delete Session in Manager View**

Project Name: Tripmaker: A Travel Agency Management System	Test Designed by: S M Abid Hassan
Test Case ID: TPM-MANAGER_VIEW-TC14	Test Designed date: 6/5/25
Test Priority (Low, Medium, High): High	Test Executed by:
Module Name: Delete Session in Manager View	Test Execution date:
Test Title: Verify Manager view functionalities (delete)	
Description: Test application Manager View pages	

Precondition (If any): Need to Log in as admin				
Test Steps	Test Data	Expected Results	Actual Result	Status(Pass/Fail)
1. Open application 2. Log in as admin to view Homepage 3. Click Go button beside a specific manager view 4. Select the Information to delete 5. Click Delete button	Select from Data Table	Data has been deleted successfully.		
		Error message will be shown.		
Post Condition: Selected data will be deleted from the database.				

## 8. ITEM PASS/FAIL CRITERIA

The following criteria will be used to determine whether individual test items (features or modules) have passed or failed during the execution of test cases in the TripMaker system:

### Pass Criteria

A test item will be considered Passed if:

- All associated test cases for the item execute without errors and produce the expected output.
- The item meets the functional requirements as defined in the Software Requirements Specification (SRS).
- No critical or high-severity defects are found in the item.
- For non-functional requirements (performance, security, usability), the item meets acceptable thresholds defined in the test plan.
- The item is successfully integrated with dependent modules and passes relevant integration tests.

### Fail Criteria

A test item will be considered Failed if:

- One or more test cases result in incorrect or unexpected behavior.
- Any critical or high-severity defects remain unresolved at the time of test completion.
- The item does not meet the specified acceptance criteria or functional requirements.
- The test item causes system crashes, data loss, or unauthorized access.
- The item fails to interact correctly with other modules in integration or system testing.

Items that partially meet the criteria (e.g., pass with low-priority issues) may be flagged for retesting or marked as conditionally passed, pending business approval.

## 9. TEST DELIVERABLES

During and after the testing process, the following documents and materials will be delivered to ensure that testing activities are traceable, transparent, and complete. These deliverables serve as formal records for validation, defect tracking, and audit purposes.

### List of Test Deliverables:

1. **Test Plan Document**
  - Outlines the overall testing strategy, scope, objectives, resources, test items, schedule, and risk analysis for the TripMaker system.
2. **Test Case Specification**
  - A detailed list of test cases, including test case IDs, descriptions, inputs, expected outputs, and status fields for pass/fail tracking.
3. **Test Scripts (Manual/Automated)**
  - Manual step-by-step scripts for UI and functional testing, and any Selenium or Postman automation scripts used in testing the application.
4. **Test Data**
  - Input datasets created specifically to validate features like registration, booking, profile updates, and payment processes. Includes both valid and invalid data samples.
5. **Test Execution Reports**
  - Contains the results of all executed test cases with pass/fail status, remarks, and defect references if applicable.
6. **Defect Report / Bug Log**
  - A record of all identified bugs, including severity, status, module affected, reproduction steps, and resolution details.
7. **Test Summary Report**
  - A high-level summary of the overall testing effort, including coverage statistics, defect trends, critical issues, and test outcome.
8. **User Acceptance Test (UAT) Report**
  - A document signed off by users/stakeholders summarizing the results of acceptance testing, confirming that the system meets business requirements.
9. **Meeting Notes and Communication Records**



- Agendas, decisions, and action items from testing meetings to maintain accountability and coordination.
10. **Configuration & Environment Details**
- Information about the hardware/software environments, versions, and tools used during testing to ensure reproducibility.
11. **Test Deliverables Handoff Checklist**
- A final checklist confirming all deliverables have been created, reviewed, and handed off to stakeholders before project closure.

## 10. STAFFING AND TRAINING NEEDS

For ensuring the project quality for this project, we need to build a good testing team. There are three common ways to recruit and train the team: **Horizontal**, **Vertical**, and **Mixed** methods.

- **Horizontal Method:**  
In Horizontal model tester's need to perform one kind of testing for many different products within an organization. This testers plays very important role of an organization. So, We hire new testers who already have experience.

### **How to recruit:**

- Post job ads on job websites.
- Contact recruitment agencies.
- Look for people who already work experience as a testers in other organizations.
- Select those people who has skills in testing and knowledge of travel-related systems.

### **Training:**

- Give an overview of the project.
  - Explain them how our system will work.
  - Show them our testing process and tools which is used in the project.
- **Vertical Model:**  
In vertical model we need tester who would recognize around the project, where dedicated people perform one or more testing tasks for the project. For this testing we choose people from inside the company (like junior developers or analysts) and train them because they already know how the company works.

### **How to recruit:**

- Look at current employees like junior developers, analysts or support staff.
- Choose those who are interested in testing or learning new skills.
- Promote or move them into testing roles.

### **Training:**

- Teach them how to test software step-by-stem.
- Give the hands-on practice with tools like Selenium, Postman or JIRA.

- Pair them with experienced tester for support.
- **Mixed Model:**  
Mixed Model method used in large software organization that combine both Horizontal and Vertical model together in the testing process. For ensuring the product quality and security we need testers who know vertical and horizontal both. In this model Experienced tester(Horizontal) helps the internal team.

**How to recruit:**

- Hire some experienced testers from outside(Horizontal) and also select and train from inside the company(Vertical).

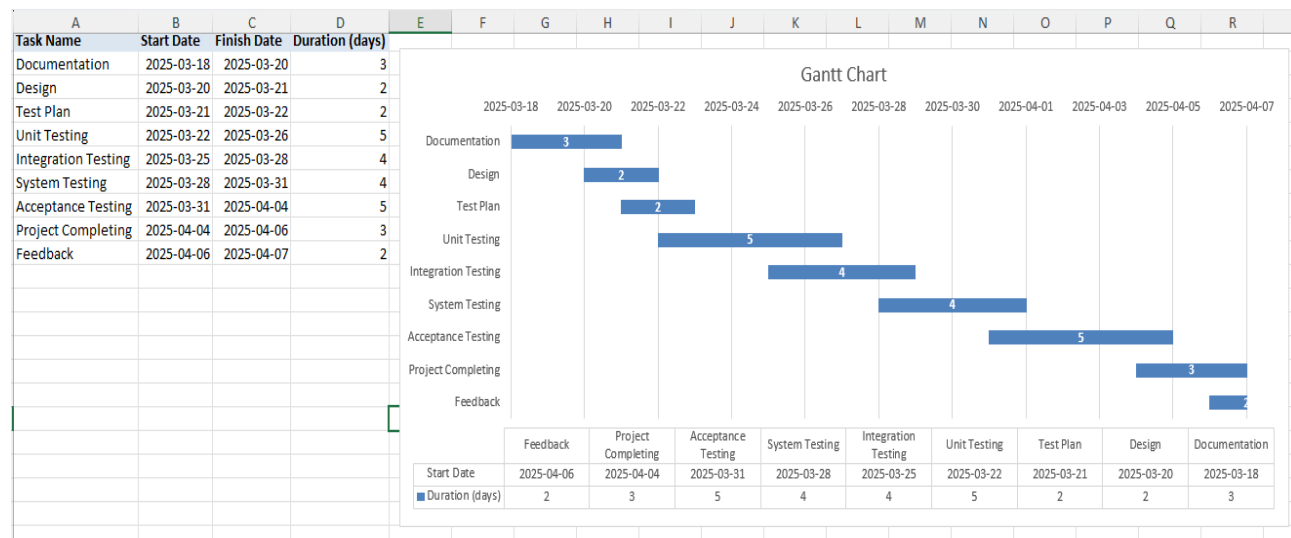
**Training:**

- Explain our project and goals.
- Give group training sessions for tools, processes, and project flow.
- Organize regular practice task, discussion and review.

## 11. RESPONSIBILITIES

Stakeholder	Role	Responsibilities in Testing
Project Manager	Manages the whole project	Approves testing plan and schedules. Estimate time, budget and support.
Test Leader(QA Lead)	Leads the testing team	Creates test strategy and test plan, give tasks to testers and check test progress.
Testers(QA Team)	Test the system	Write and run test cases, Find and reports bugs.
Developers	Build the software	Fix bugs found by testers, Work with QA for smooth testing(Unit Testing)
Business Analyst	Defines requirements	Explains business rules, Validates that test cover all business and customer needs.
UI/UX Designer	Designs user interface	Ensures design matches test result and user expectations.
Client/End Users	Review and approve features	Participate in user acceptance testing. Give feedback on features and usability.
System Administrator	Maintains test environment	Maintenance, configuration, and reliable operation of computer systems and servers.

## 12. TESTING SCHEDULE



## 13. PLANNING RISKS AND CONTINGENCIES

Table 1: Risk Mitigation Plan for testing

S/N	Risk Description	Possibility	Impact	Mitigation Plan
1	Integration of multiple modules	70%	2	Implement modular development to manage complexity effectively
2	Inclusion of extensive features	50%	2	Prioritize features based on user needs and feasibility assessments
3	Incorporating complex algorithms for itinerary optimization	30%	3	Conduct thorough algorithm testing and optimization cycles during development
4	Fluctuating market demand	60%	1	Maintain a contingency fund
5	Competitor actions disrupting market dynamics	40%	2	Regularly monitor competitor strategies and adjust project plans accordingly

6	Changing regulatory requirements	20%	4	Establish a compliance team to stay updated on regulatory changes and adapt the system accordingly.
7	Customers expecting advanced features	60%	3	Conduct user research to understand customer expectations and preferences.
8	Customers from diverse cultural backgrounds	20%	3	Hire multilingual staff or utilize translation services for effective communication.
9	Managing time zone differences	30%	4	Implement asynchronous communication tools and establish clear response time expectations.
10	Lack of adherence to defined Agile practices	70%	2	Provide regular Agile training sessions and conduct frequent process audits.
11	Inadequate documentation	40%	3	Implement a robust documentation management system and enforce documentation standards.
12	Resistance to change	50%	3	Facilitate change management workshops and provide ongoing support to teams.
13	Limited availability of travel-specific development tools	60%	2	Invest in tool procurement and consider custom tool development.
14	Incompatibility issues between different software components	80%	2	Conduct thorough compatibility testing.
15	Insufficient infrastructure for testing	60%	1	Scale up testing infrastructure
16	Integration of emerging technologies like AI	40%	3	Collaborate with AI experts

17	Implementing blockchain for secure transactions and data integrity.	20%	4	Engage blockchain specialists
18	Incorporating IoT for real- time tracking of travel assets.	30%	4	Conduct extensive feasibility studies and establish protocols
19	Limited availability of experienced developers	70%	2	Provide industry-specific training to existing staff.
20	High turnover rate	40%	2	Implement knowledge transfer sessions and mentorship programs

Table 2: Contingency Plans for Identified Risks

S/N	Risk Description	Contingency Plan
1	Integration of multiple modules	Delay dependent features and perform manual integration step-by-step
2	Inclusion of extensive features	Freeze non-critical features and shift to future updates or versions
3	Complex algorithms for itinerary optimization	Replace with simpler fallback algorithms or offer manual override
4	Fluctuating market demand	Temporarily scale down marketing or operations; adapt pricing strategy
5	Competitor actions disrupting market dynamics	Revise business model and offer targeted promotions to retain users
6	Changing regulatory requirements	Put temporary feature holds; fast-track emergency compliance updates
7	Customers expecting advanced features	Launch features in phases and provide a roadmap to users
8	Customers from diverse cultural backgrounds	Use visual UI aids and multilingual support widgets as stopgap
9	Managing time zone differences	Assign region-based testers and document async response timelines
10	Lack of adherence to Agile practices	Assign Agile coaches; enforce daily stand-ups and retrospectives
11	Inadequate documentation	Create living docs using collaborative tools like Notion or Confluence

12	Resistance to change	Assign change champions; enforce adoption through leadership involvement
13	Limited availability of travel-specific dev tools	Use general-purpose tools temporarily with workarounds
14	Incompatibility between software components	Isolate incompatible modules and rebuild interfaces incrementally
15	Insufficient testing infrastructure	Use cloud-based or shared testing environments temporarily
16	Integration of emerging technologies (AI)	Disable experimental features and focus on core functionality
17	Implementing blockchain for secure transactions	Postpone integration and use existing secure methods temporarily
18	Incorporating IoT for real-time travel asset tracking	Implement polling-based tracking or third-party APIs as interim solution
19	Limited availability of experienced developers	Outsource critical tasks or use freelancers temporarily
20	High turnover rate	Maintain updated documentation and assign shadow developers for all key roles

## 14. APROVALS

The following personnel are responsible for reviewing and approving this test plan. Their approval confirms that the test plan is accurate, complete, and ready to be used during the testing phase of the Project.

Name	Designation	Role in Approval	Signature
<b>Sha Mohamad Yeahia Idris</b>	Project Manager	Final approval of the overall test plan	
<b>Dip Achorjee Shokal</b>	QA/Test Lead	Approves test strategy, scope, and execution process	
<b>S M Abid Hasan</b>	Lead Developer	Approves test strategy, scope, and execution process	
<b>Md. Rezwan Mujahid Rudro</b>	Business Analyst	Approves test strategy, scope, and execution process	
<b>Sha Mohamad Yeahia Idris</b>	Client Representative	Approves test strategy, scope, and execution process	