

Travel Management System

Capstone Project

Case Study

By

RASHMI R

INTRODUCTION

The **Travel Management System** is a web-based application designed to automate and simplify the process of managing travel packages, hotel bookings, and user reservations.

It provides a centralized platform where **users can browse travel packages, book trips, and track booking status**, while **administrators manage packages, hotels, and booking approvals**.

The system reduces manual work, improves accuracy, and ensures a smooth booking workflow between users and administrators.

This project is developed using **Spring Boot, MySQL, HTML, CSS, JavaScript, and REST APIs**.

ABSTRACT

The **Travel Management System (TMS)** is developed to manage travel-related activities such as package creation, hotel management, user bookings, and approval workflows.

The system allows users to **register, log in, view packages, book trips, and monitor booking status**, while administrators can **add hotels, create packages, approve or cancel bookings, and monitor system statistics**.

The main goal of this project is to demonstrate **real-world CRUD operations, role-based access control, and client-server interaction** using Spring Boot and MySQL. The project follows a layered architecture consisting of **Controller, Service, DAO, and Database layers**.

CLIENT REQUIREMENTS

The client requires a Travel Management System with the following features:

- Admin should be able to:
 - Add, update, delete, and view travel packages
 - Add and manage hotels
 - View all bookings
 - Approve or cancel user bookings
 - View dashboard statistics (total users, bookings, hotels, packages)
- User should be able to:
 - Register and log in
 - View available travel packages
 - Book travel packages
 - View booking history and status
 - Cancel bookings (before approval)

TECHNICAL FEATURES

- Proper **Login and Registration**
- **Role-based authentication** (Admin / User)
- **REST APIs** for data communication
- CRUD operations on:
 - ✓ Users
 - ✓ Hotels
 - ✓ Travel Packages
 - ✓ Bookings
- Booking approval workflow

- Dashboard with real-time statistics
- Exception handling and validation
- Clean UI with HTML, CSS, and JavaScript

TECHNOLOGIES AND TOOLS USED

- ✓ **Programming Language:** Java (JDK 17)
- ✓ **Framework:** Spring Boot
- ✓ **Frontend:** HTML, CSS, JavaScript
- ✓ **Database:** MySQL 8.0
- ✓ **IDE:** Spring Tool Suite (STS)
- ✓ **Web Server:** Apache Tomcat (embedded)
- ✓ **Build Tool:** Maven
- ✓ **API Testing:** Postman
- ✓ **Browser:** Google Chrome

PROJECT MODULES

- ❖ **Admin Module**
- ❖ **User Module**
- ❖ **Hotel Module**
- ❖ **Travel Package Module**
- ❖ **Booking Module**

MODULE DESCRIPTION

1. Admin Module

The **Admin Module** is responsible for managing and monitoring the entire Travel Management System. It provides administrative control over users, travel packages, hotels, and bookings.

Features:

- Admin login with authentication
- Dashboard overview showing:
 - Total users
 - Total hotels
 - Total travel packages
 - Total bookings
- Add new travel packages
- Add hotel details
- View all user bookings
- Approve or cancel user bookings
- Monitor pending bookings

Purpose:

This module ensures that all system activities are properly controlled and managed. Only authorized admin users can access this module.

2. User Module

The **User Module** allows customers to interact with the system, explore travel options, and make bookings.

Features:

- User registration
- User login and logout
- View available travel packages
- Book a travel package
- View personal booking history
- Check booking status (BOOKED / APPROVED / CANCELLED)
- Cancel booking (if booking is still pending)

Purpose:

This module provides a smooth interface for users to plan and manage their travel bookings.

3. Travel Package Module

The **Travel Package Module** manages all travel packages offered by the system.

Features:

- Add new travel packages (Admin only)
- View list of all travel packages
- Display package details:
 - Destination
 - Price
 - Number of days
 - Description
 - Associated hotel
- Delete travel packages (Admin only)

Purpose:

This module acts as the core of the application by defining travel plans that users can browse and book.

4. Hotel Module

The **Hotel Module** manages hotel details linked to travel packages.

Features:

- Add hotel details (Admin only)
- Store hotel information such as:
 - Hotel name
 - Location
 - Price per day
- Associate one hotel with a travel package
- Display hotel details in bookings

Purpose:

This module ensures that each travel package is associated with proper accommodation details, enhancing the realism of the system.

5. Booking Module

The **Booking Module** handles the entire booking workflow between users and administrators.

Features:

- User books a travel package
- Booking status initially set to **BOOKED**
- Admin reviews and updates booking status:

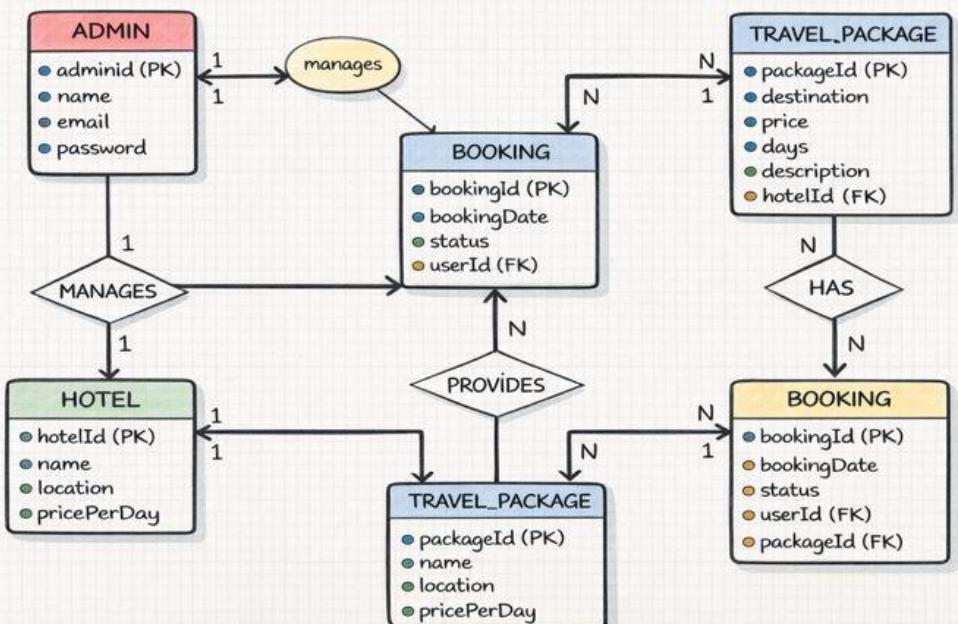
- APPROVED
 - CANCELLED
- User views booking history and status
 - Admin views all bookings and pending requests

Purpose:

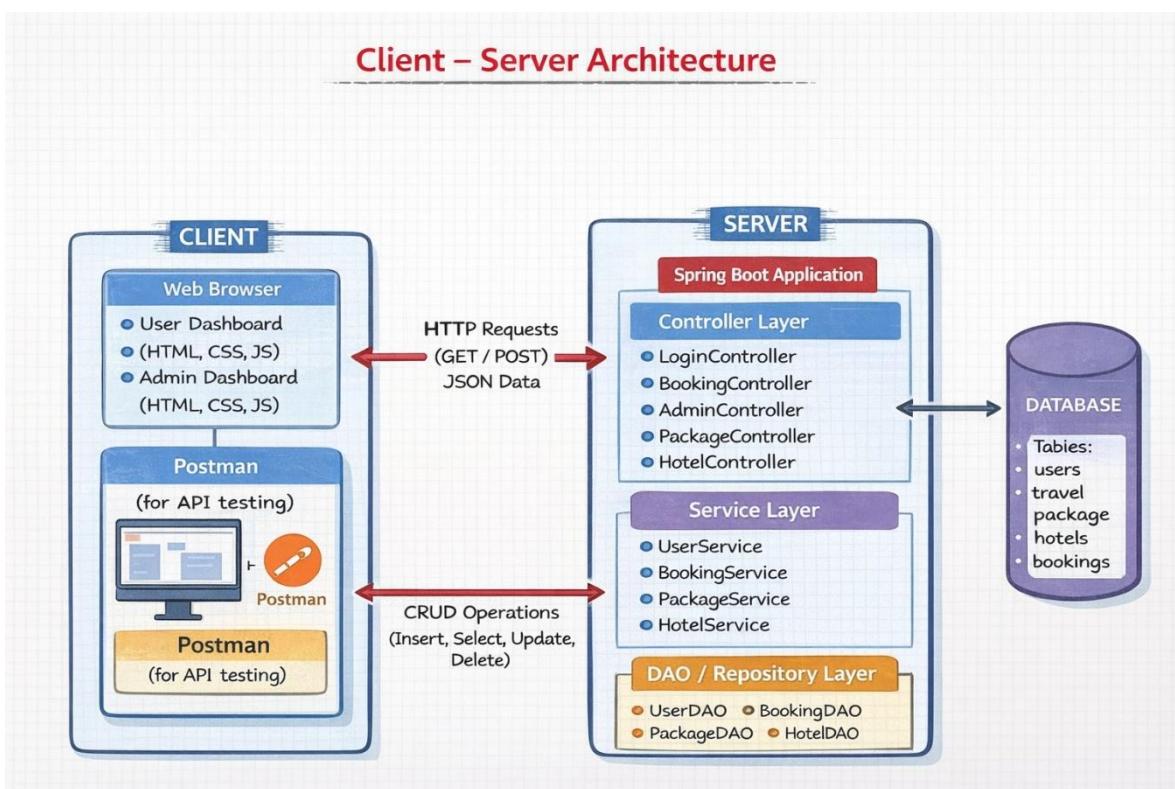
This module manages the interaction between users and admins, ensuring bookings are properly tracked and approved.

ER DIAGRAM

ER DIAGRAM



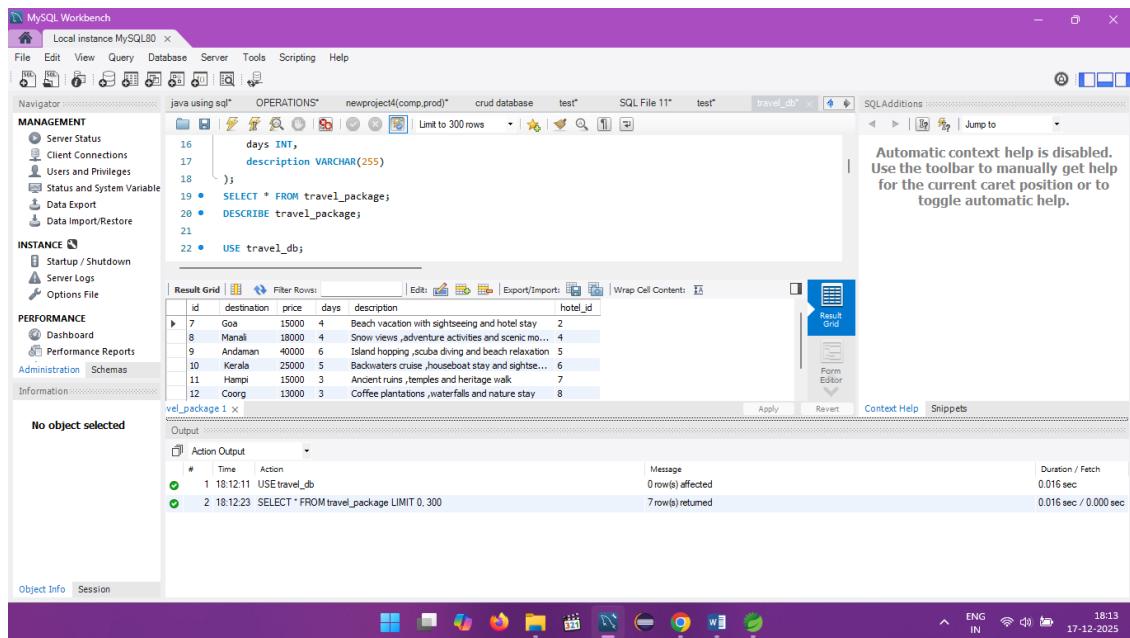
CLIENT-SERVER ARCHITECTURE



DATA DICTIONARY

TABLES OF SQL DATABASES

The Travel Management System uses MySQL as the backend database. The database consists of four main tables: Users, Hotels, Travel Packages, Bookings. These tables are connected using foreign key relationships to maintain data integrity.



```
java using sql*    OPERATIONS*  newproject4(comp.prod)*  crud database  test*  SQL File 11*  test*  travel_db*  SQLAdditions...  SQL Editor  Jump to  MANAGEMENT  INSTANCE  PERFORMANCE  Administration  Information  No object selected  vel_package1.x  Object Info  Session  Output  Action Output  Time  Action  Message  Duration / Fetch  1 18:12:11 USEtravel_db  0row(s) affected  0.016 sec  2 18:12:23 SELECT * FROM travel_package LIMIT 0, 300  7row(s) returned  0.016 sec / 0.000 sec  18:13  ENG IN  17-12-2025
```

| | id | destination | price | days | description | hotel_id |
|----|---------|-------------|-------|---|-------------|----------|
| 7 | Goa | 15000 | 4 | Beach vacation with sightseeing and hotel stay | 2 | |
| 8 | Manali | 18000 | 4 | Snow views, adventure activities and scenic mo... | 4 | |
| 9 | Andaman | 40000 | 6 | Island hopping, scuba diving and beach relaxation | 5 | |
| 10 | Kerala | 25000 | 5 | Backwaters cruise, houseboat stay and sightsee... | 6 | |
| 11 | Hampi | 15000 | 3 | Ancient ruins, temples and heritage walk | 7 | |
| 12 | Coorg | 13000 | 3 | Coffee plantations, waterfalls and nature stay | 8 | |

Travel-Packages Table

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: Local instance MySQL80

MANAGEMENT: Server Status, Client Connections, Users and Privileges, Status and System Variable, Data Export, Data Import/Restore

INSTANCE: Startup / Shutdown, Server Logs, Options File

PERFORMANCE: Dashboard, Performance Reports

Administration: Schemas

Information: No object selected

Result Grid: users

| ID | Name | Email | Password | Role |
|----|-----------|-------------------|----------|-------|
| 1 | Admin | admin@gmail.com | admin123 | ADMIN |
| 5 | Test User | user@gmail.com | user123 | USER |
| 6 | Rashmi R | rashmi@gmail.com | 12345 | USER |
| 12 | Kashika | kashika@gmail.com | 1216 | USER |

Output:

| # | Time | Action | Message | Duration / Fetch |
|---|----------|---|-------------------|-----------------------|
| 1 | 18:12:11 | USE travel_db | 0 row(s) affected | 0.016 sec |
| 2 | 18:12:23 | SELECT * FROM travel_package LIMIT 0, 300 | 7 row(s) returned | 0.016 sec / 0.000 sec |
| 3 | 18:14:56 | SELECT * FROM users LIMIT 0, 300 | 4 row(s) returned | 0.000 sec / 0.000 sec |

Object Info Session

18:15 ENG IN 17-12-2025

Users Table

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator: Local instance MySQL80

MANAGEMENT: Server Status, Client Connections, Users and Privileges, Status and System Variable, Data Export, Data Import/Restore

INSTANCE: Startup / Shutdown, Server Logs, Options File

PERFORMANCE: Dashboard, Performance Reports

Administration: Schemas

Information: No object selected

Result Grid: hotels

| ID | Name | Location | Price_per_day |
|----|---------------------|----------|---------------|
| 1 | Taj Residency | Goa | 4500 |
| 2 | Taj Residency | Goa | 3500 |
| 3 | Taj Residency | Goa | 3500 |
| 4 | Snow Peak Inn | Manali | 4000 3500 |
| 5 | Coral Island Resort | Andaman | 5500 |

Output:

| # | Time | Action | Message | Duration / Fetch |
|---|----------|--|---|-----------------------|
| 1 | 18:12:11 | USE travel_db | 0 row(s) affected | 0.016 sec |
| 2 | 18:12:23 | SELECT * FROM travel_package LIMIT 0, 300 | 7 row(s) returned | 0.016 sec / 0.000 sec |
| 3 | 18:14:56 | SELECT * FROM users LIMIT 0, 300 | 4 row(s) returned | 0.000 sec / 0.000 sec |
| 4 | 18:15:29 | INSERT INTO users(name, email, password) VALUES ('Admin', 'admin@gmail.com', 'admin123') | Error Code: 1364. Field 'role' doesn't have a default value | 0.000 sec |
| 5 | 18:16:16 | SELECT * FROM hotels LIMIT 0, 300 | 9 row(s) returned | 0.000 sec / 0.000 sec |

Object Info Session

18:16 ENG IN 17-12-2025

Hotels Table

The screenshot shows the MySQL Workbench interface. The left sidebar includes sections for MANAGEMENT (Server Status, Client Connections, Users and Privileges, Status and System Variable, Data Export, Data Import/Restore), INSTANCE (Startup / Shutdown, Server Logs, Options File), PERFORMANCE (Dashboard, Performance Reports), Administration (Schemas), and Information. A message at the top states: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help." The central workspace displays a schema editor with the following code:

```
25* java using sql* OPERATIONS* newproject4(comp.prod)* crud database* test* SQL File 11* test* travel_db* x SQLAdditions... Limit to 300 rows
76    status VARCHAR(20) DEFAULT 'BOOKED',
77
78    FOREIGN KEY (package_id) REFERENCES travel_package(id)
79  );
80
81 • SELECT * FROM bookings;
82
```

Below the code is a Result Grid showing the data from the bookings table:

| | id | user_email | package_id | booking_date | status |
|---|----|----------------|------------|---------------------|----------|
| ▶ | 1 | user@gmail.com | 1 | 2025-12-15 11:23:00 | APPROVED |
| | 2 | user@gmail.com | 3 | 2025-12-15 11:23:17 | BOOKED |
| | 3 | user@gmail.com | 2 | 2025-12-15 11:23:46 | APPROVED |
| | 4 | user@gmail.com | 1 | 2025-12-15 11:56:17 | BOOKED |
| | 5 | user@gmail.com | 1 | 2025-12-15 11:58:52 | BOOKED |

The bottom section shows the Output pane with two log entries:

| Action | Time | Action | Message | Duration / Fetch |
|--------|----------|-------------------------------------|--------------------|-----------------------|
| 1 | 18:19:12 | USE travel_db | 0 row(s) affected | 0.000 sec |
| 2 | 18:19:24 | SELECT * FROM bookings LIMIT 0, 300 | 29 row(s) returned | 0.000 sec / 0.000 sec |

Bookings Table

The screenshot shows the MySQL Workbench interface. The left sidebar includes sections for MANAGEMENT (Server Status, Client Connections, Users and Privileges, Status and System Variable, Data Export, Data Import/Restore), INSTANCE (Startup / Shutdown, Server Logs, Options File), PERFORMANCE (Dashboard, Performance Reports), Administration, and Schemas. The central area has tabs for Navigator, File, Edit, View, Query, Database, Server, Tools, Scripting, Help, and a toolbar with various icons. A query editor window is open, showing the following SQL code:

```
25*    java using sql* OPERATIONS* newproject4(comp.prod)* crud database* test* SQL File 11* test* travel_db* SQLAdditions* Limit to 300 rows
85
86 ● DESCRIBE bookings;
87
88
89 ● SHOW TABLES;
90
91 ● ALTER TABLE bookings
```

The Result Grid shows the following table structure:

| Tables_in_travel_db |
|---------------------|
| bookings |
| hotels |
| travel_package |
| users |

The bottom pane shows the Output log with the following entries:

| # | Time | Action | Message | Duration / Fetch |
|---|----------|-------------------------------------|--------------------|-----------------------|
| 1 | 18:19:12 | USE travel_db | 0 row(s) affected | 0.000 sec |
| 2 | 18:19:24 | SELECT * FROM bookings LIMIT 0, 300 | 29 row(s) returned | 0.000 sec / 0.000 sec |
| 3 | 18:19:40 | DESCRIBE bookings | 5 row(s) returned | 0.000 sec / 0.000 sec |
| 4 | 18:19:50 | SHOW TABLES | 4 row(s) returned | 0.016 sec / 0.000 sec |

A context help message is displayed on the right: "Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help."

Tables in travel_db

HTTP Request Methods

Definition:

HTTP defines a set of request methods to indicate the action to be performed on a given resource.

| HTTP Method | URI | Purpose |
|-------------|---|-------------------------------|
| GET | http://localhost:8088/view-packages | Fetch all travel packages |
| GET | http://localhost:8088/bookPackage?packageId=3 | Book a selected package |
| GET | http://localhost:8088/my-bookings | Retrieve user booking history |
| POST | http://localhost:8088/doLogin | Login authentication |
| POST | http://localhost:8088/register | Register new user |
| POST | http://localhost:8088/addPackage | Create new travel package |
| POST | http://localhost:8088/addHotel | Create new hotel |
| DELETE | http://localhost:8088/deletePackage?id=7 | Delete travel package |

URI (Uniform Resource Identifier)

Definition

URI is used to uniquely identify resources in a web application. In this project, URI is used to identify users, travel packages, hotels, and bookings.

| HTTP Method | URI | Description |
|-------------|---|--------------------------------------|
| GET | http://localhost:8088/login | Display login page |
| GET | http://localhost:8088/register | Display registration page |
| GET | http://localhost:8088/logout | Logout from system |
| GET | http://localhost:8088/admin-dashboard | Display admin dashboard |
| GET | http://localhost:8088/user-dashboard | Display user dashboard |
| GET | http://localhost:8088/view-packages | View all travel packages |
| GET | http://localhost:8088/my-bookings | View logged-in user bookings |
| GET | http://localhost:8088/add-package | Open add travel package page (Admin) |
| POST | http://localhost:8088/addPackage | Add new travel package |
| GET | http://localhost:8088/add-hotel | Open add hotel page (Admin) |
| POST | http://localhost:8088/addHotel | Add new hotel |
| GET | http://localhost:8088/bookPackage?packageId=5 | Book a travel package |
| POST | http://localhost:8088/admin/update-booking | Approve/Reject booking |

POSTMAN DEMONSTRATION

The screenshot shows the Postman interface with a successful API call. The URL is `http://localhost:8088/api/packages`. The response body is a JSON array of package objects:

```
1 [ 2 { 3 "days": 4, 4 "description": "Beach vacation with sightseeing and hotel stay", 5 "destination": "Goa", 6 "hotelId": 2, 7 "hotelLocation": "Goa", 8 "hotelName": "Taj Residency", 9 "id": 7, 10 "price": 16000.0 11 }, 12 { 13 "days": 4, 14 "description": "Snow views ,adventure activities and scenic\\r\\nmountain tour", 15 "destination": "Manali", }
```

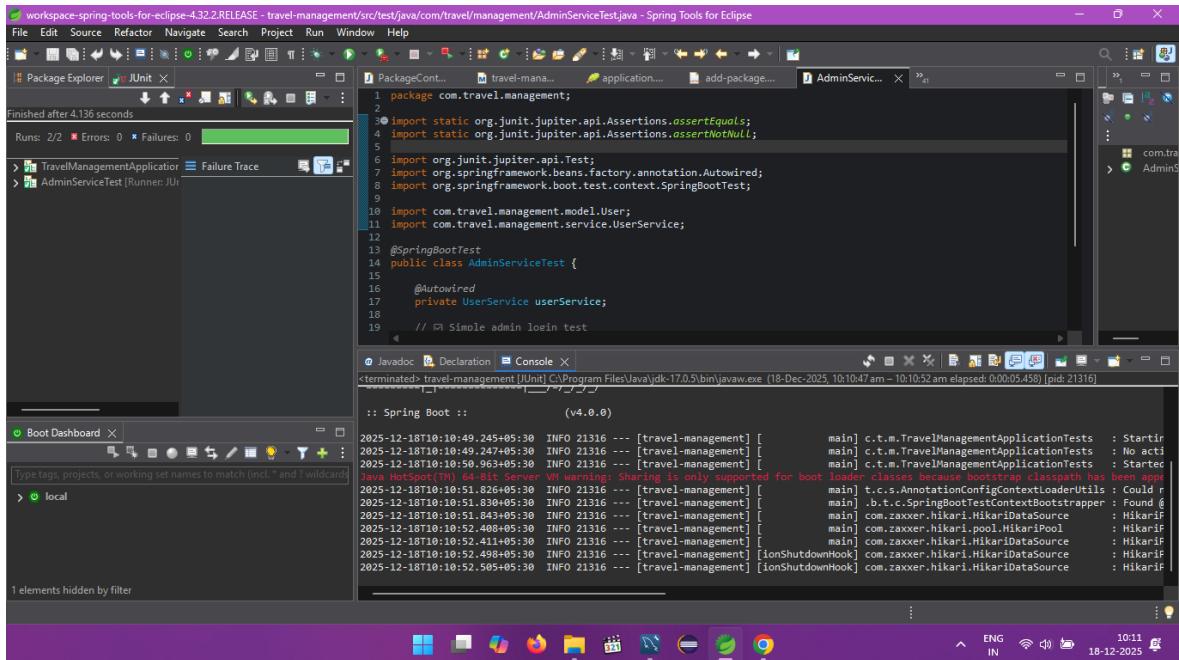
Get Method for packages

The screenshot shows the Postman interface with a successful API call. The URL is `http://localhost:8088/api/hotels`. The response body is a JSON array of hotel objects:

```
1 [ 2 { 3 "id": 1, 4 "location": "Goa", 5 "name": "Taj Residency", 6 "pricePerDay": 4500.0 7 }, 8 { 9 "id": 2, 10 "location": "Goa", 11 "name": "Taj Residency", 12 "pricePerDay": 3500.0 13 }, 14 { 15 "id": 3,
```

Get method for hotels

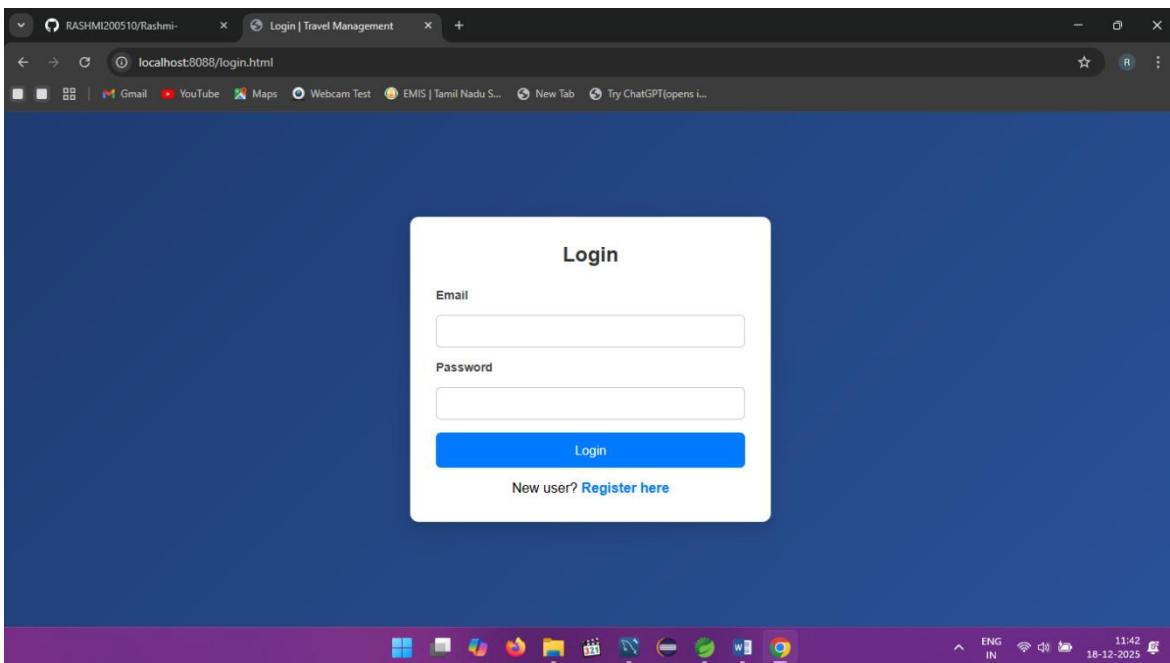
JUNIT TEST FOR ADMIN MODULE



The screenshot shows the Eclipse IDE interface with the following details:

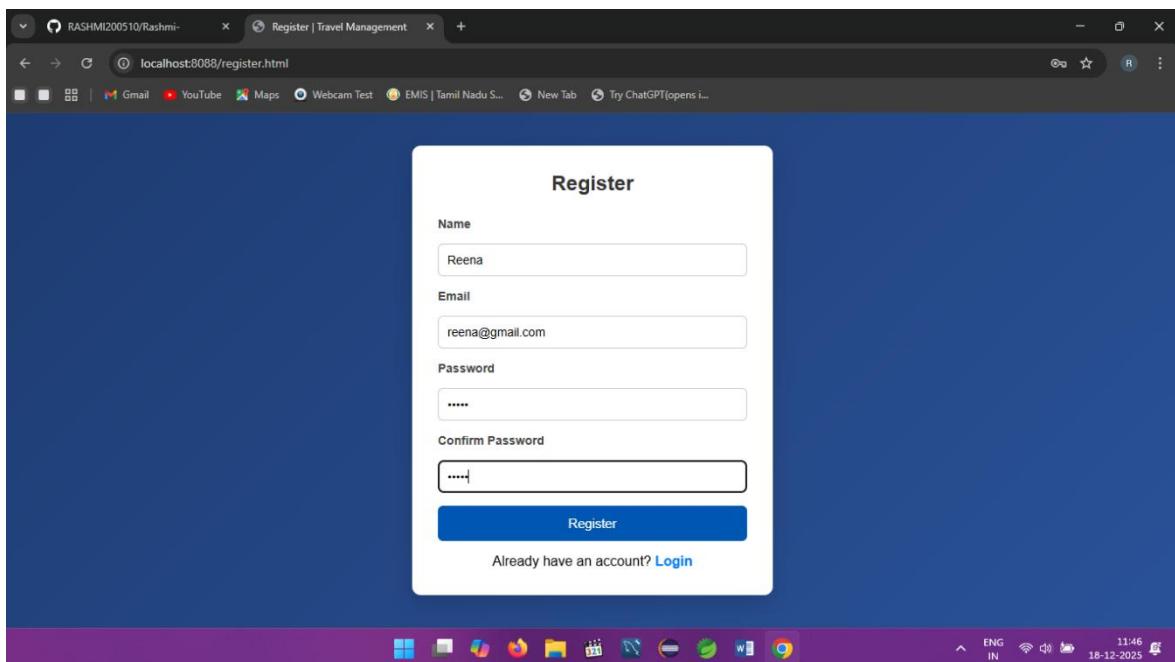
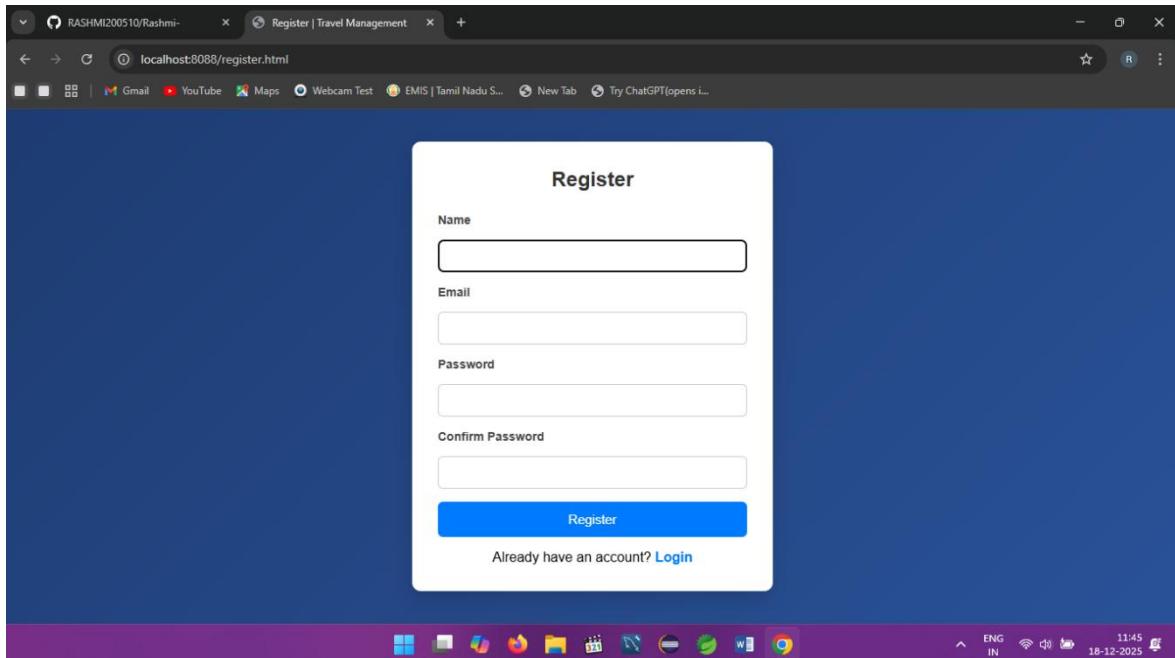
- File Bar:** File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help.
- Toolbar:** Standard Eclipse toolbar.
- Left Sidebar:** Package Explorer, JUnit (highlighted), Navigator, Problems, Search, Project, Run, Window, Help.
- Middle Area:**
 - JUnit View:** Shows "Finished after 4.136 seconds", "Runs: 2/2", "Errors: 0", and "Failures: 0".
 - Code Editor:** Displays the `AdminServiceTest.java` file content.
 - Console View:** Shows Java application logs and Spring Boot startup logs.
- Right Sidebar:** Shows the project structure with `com.travel.management` and `AdminServiceTest`.
- Bottom:** Eclipse status bar and Windows taskbar.

PROJECT OUTPUTS

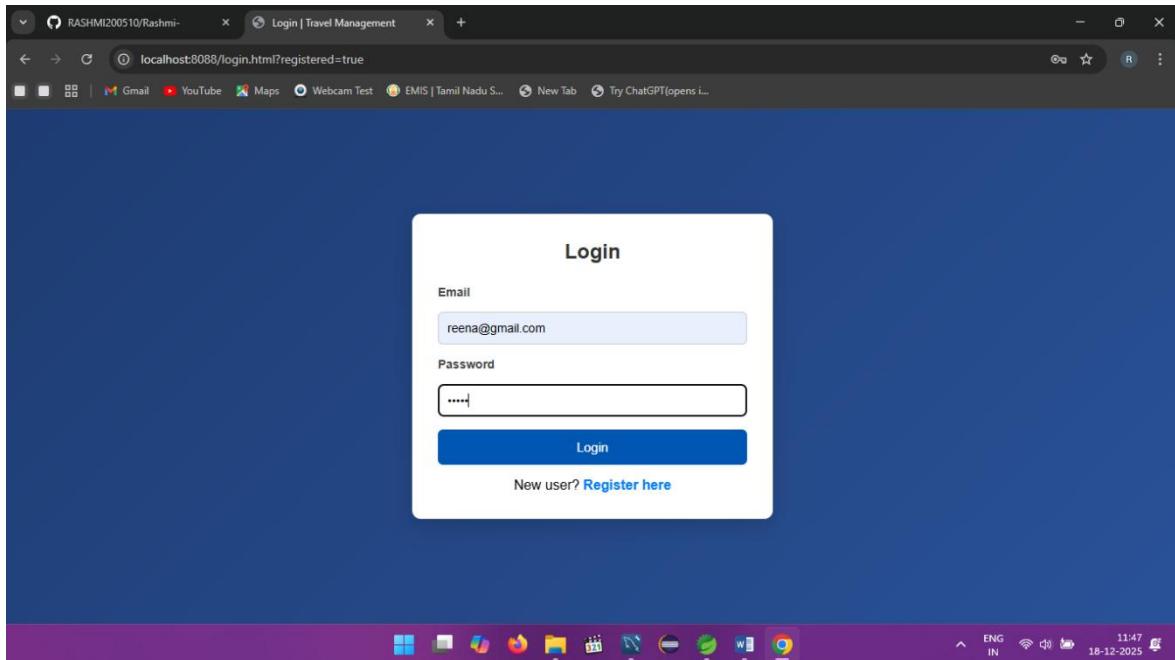


Login Page

Registration Page



User Login



User Dashboard

A screenshot of a web browser window titled "RASHMI200510/Rashmi" showing a "User Dashboard". The URL in the address bar is "localhost:8088/user-dashboard.html?sessionid=108632529CFB3511181B6D48A40F4EBA". The page has a dark blue header with "Travel Management" on the left and "Logged in as: User" with a "Logout" button on the right. A sidebar on the left lists "Dashboard", "View Packages", and "My Bookings". The main content area starts with a "Welcome User" message: "Your travel summary is shown below." Below this are three cards: "Total Bookings" (0), "Approved Trips" (0), and "Pending Approval" (0). Further down is a section titled "Approved Travel Receipts" with a table. The table has columns: Destination, Days, Price, Booking Date, and Status. The table body contains the text "No approved trips yet". The browser's toolbar and taskbar are visible at the top and bottom respectively.

Travel Packages

A screenshot of a web browser window titled "View Travel Packages". The URL in the address bar is "localhost:8088/view-packages.html". The page displays a table titled "Travel Packages" with the following data:

| ID | Destination | Price | Days | Description | Action |
|----|-------------|-------|------|---|-----------------------|
| 7 | Goa | 15000 | 4 | Beach vacation with sightseeing and hotel stay | <button>Book</button> |
| 8 | Manali | 18000 | 4 | Snow views ,adventure activities and scenic mountain tour | <button>Book</button> |
| 9 | Andaman | 40000 | 6 | Island hopping ,scuba diving and beach relaxation | <button>Book</button> |
| 10 | Kerala | 25000 | 5 | Backwaters cruise ,houseboat stay and sightseeing | <button>Book</button> |
| 11 | Hampi | 15000 | 3 | Ancient ruins, temples and heritage walk | <button>Book</button> |
| 12 | Coorg | 13000 | 3 | Coffee plantations ,waterfalls and nature stay | <button>Book</button> |
| 14 | Rajasthan | 18000 | 4 | Heritage hotel with guided city tour and cultural events. | <button>Book</button> |

My Booked Travel Packages

A screenshot of a web browser window titled "My Bookings". The URL in the address bar is "localhost:8088/my-bookings.html?success=true". The page displays a message "Booking completed successfully!" and a button "Go to Dashboard / View Receipt". Below this, there is a table titled "My Booked Travel Packages" with the following data:

| Destination | Hotel | Location | Days | Price | Booking Date | Status | Action |
|-------------|-------------------------------|----------|------|-------|---------------------|--------|-------------------------|
| Coorg | Misty Hills Plantation Resort | Coorg | 3 | 13000 | 2025-12-18 11:48:32 | BOOKED | <button>Cancel</button> |

User Dashboard After Booking

The screenshot shows a user dashboard for a travel management system. The top navigation bar includes links for Gmail, YouTube, Maps, Webcam Test, EMIS | Tamil Nadu S..., New Tab, and Try ChatGPT. The main content area has a dark blue sidebar on the left with options: Dashboard, View Packages, and My Bookings. The main panel displays a "Welcome User" message and a summary of travel data: 1 Total Bookings, 0 Approved Trips, and 1 Pending Approval. Below this is a section titled "Approved Travel Receipts" with a table header and a message stating "No approved trips yet". The bottom right corner shows system status: ENG IN, 11:48, 18-12-2025.

Admin Login

The screenshot shows an admin login page with a dark blue background. The page title is "Login" and it contains fields for "Email" (admin@gmail.com) and "Password" (represented by a redacted box). A blue "Login" button is at the bottom. Below the button, text says "New user? [Register here](#)". The bottom right corner shows system status: ENG IN, 11:49, 18-12-2025.

Admin Dashboard

The screenshot shows the Admin Dashboard for a Travel Management system. The top navigation bar includes links for Gmail, YouTube, Maps, Webcam Test, EMIS | Tamil Nadu S..., New Tab, and Try ChatGPT. The dashboard header displays "Logged in as: Admin" and a "Logout" button. On the left, a sidebar menu lists "Travel Management" sections: Dashboard, Travel Packages, Add Package, and Add Hotel. The main content area features a "Dashboard Overview" section with four cards: "Total Bookings" (33), "Total Packages" (7), "Total Hotels" (9), and "Total Users" (6). Below this is a "Pending Bookings" table:

| User | Destination | Hotel | Days | Price | Status | Action |
|-----------------|-------------|-------------------------------|------|-------|--------|--|
| reena@gmail.com | Coorg | Misty Hills Plantation Resort | 3 | 13000 | BOOKED | <button>Approve</button> <button>Reject</button> |

The taskbar at the bottom shows various application icons and the system clock indicating 11:49 AM on 18-12-2025.

Bookings Page of Admin

The screenshot shows the "All Bookings" page. The top navigation bar includes links for Gmail, YouTube, Maps, Webcam Test, EMIS | Tamil Nadu S..., New Tab, and Try ChatGPT. The main content area displays a table of bookings:

| User | Destination | Days | Price | Date | Status | Action |
|-------------------|-------------|------|-------|---------------------|-----------|--------|
| user@gmail.com | Coorg | 3 | 13000 | 2025-12-16 12:03:33 | APPROVED | - |
| user@gmail.com | Manali | 4 | 18000 | 2025-12-16 12:24:23 | APPROVED | - |
| user@gmail.com | Hampi | 3 | 15000 | 2025-12-16 14:33:38 | CANCELLED | - |
| user@gmail.com | Andaman | 6 | 40000 | 2025-12-16 18:28:56 | APPROVED | - |
| user@gmail.com | Goa | 4 | 15000 | 2025-12-16 19:42:43 | APPROVED | - |
| user@gmail.com | Manali | 4 | 18000 | 2025-12-16 22:20:50 | APPROVED | - |
| user@gmail.com | Kerala | 5 | 25000 | 2025-12-16 22:22:56 | APPROVED | - |
| kashika@gmail.com | Goa | 4 | 15000 | 2025-12-16 22:27:02 | APPROVED | - |
| user@gmail.com | Kerala | 5 | 25000 | 2025-12-17 19:58:24 | APPROVED | - |
| user@gmail.com | Goa | 4 | 15000 | 2025-12-17 20:41:49 | APPROVED | - |
| harshi@gmail.com | Kerala | 5 | 25000 | 2025-12-18 10:17:02 | APPROVED | - |
| reena@gmail.com | Coorg | 3 | 13000 | 2025-12-18 11:48:32 | APPROVED | - |

The taskbar at the bottom shows various application icons and the system clock indicating 11:49 AM on 18-12-2025.

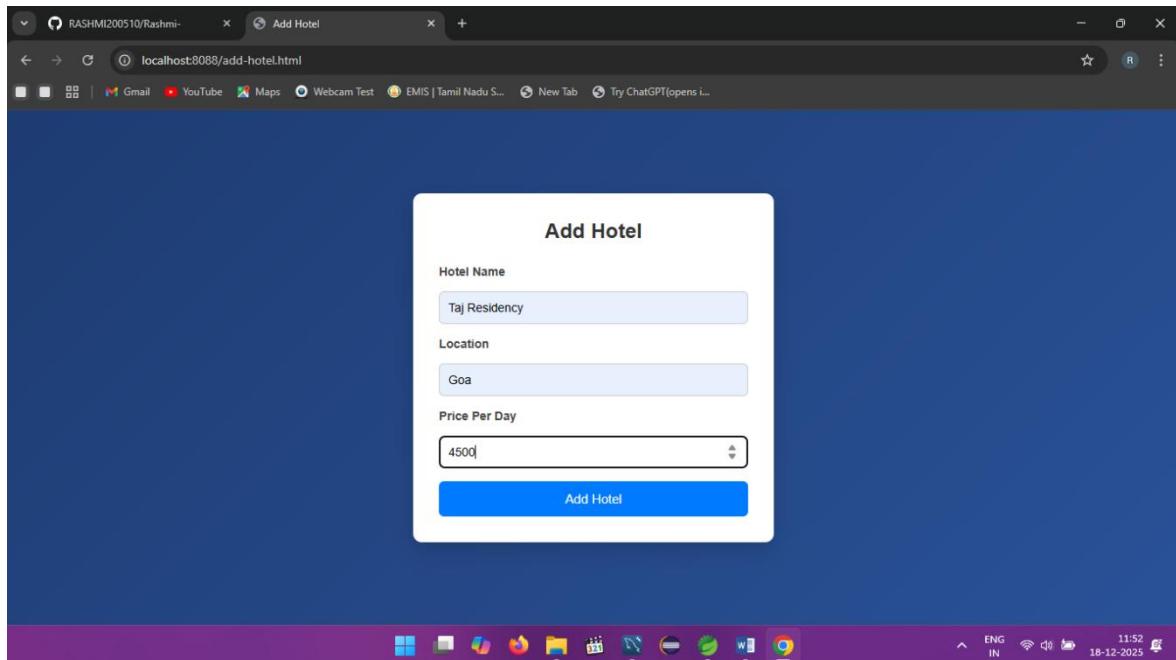
Updated Admin Dashboard

The screenshot shows a web browser window titled "Admin Dashboard" with the URL "localhost:8088/admin-dashboard.html". The page has a dark blue header bar with the text "Travel Management" on the left and "Logged in as: Admin" with a "Logout" button on the right. Below the header is a sidebar with links: "Dashboard", "Travel Packages", "Add Package", and "Add Hotel". The main content area is titled "Dashboard Overview" and contains four boxes showing statistics: "Total Bookings" (33), "Total Packages" (7), "Total Hotels" (9), and "Total Users" (6). Below this is a section titled "Pending Bookings" with a table header row and a message "No pending bookings". The bottom of the screen shows a Windows taskbar with various icons and system status indicators.

Add Travel Packages Page

The screenshot shows a web browser window titled "Add Travel Package" with the URL "localhost:8088/add-package.html". The page has a dark blue header bar with the URL and a "Back" button. The main content area is titled "Add Travel Package" and contains several input fields: "Destination" (set to "Goa"), "Price" (set to "15000"), "Days" (set to "4"), "Description" (containing the text "Beach vacation with sightseeing and hotel stay"), and a "Hotel" dropdown menu set to "Taj Residency (Goa)". At the bottom is a large blue "Add Package" button. The bottom of the screen shows a Windows taskbar with various icons and system status indicators.

Add Hotel Page



User Dashboard After Approval

