MALNAD COLEGE OF ENGINEERING

Under the auspices of M.T.E.S

(An autonomous institution under Visvesvaraya Technological University, Belgaum)
Hassan – 573201, Karnataka, India

Report on Course Title: Full Stack Development

"Travel Booking"

Submitted by team number :03

Name	USN
Preksha KN	4MC22IS078
Rashmitha	4MC23IS086
Sahana CK	4MC23IS090
Sanika PD	4MC23IS096
Thejaswini NK	4MC23IS118

Mr. Krishna Swaroop A

(Assistant Professor Dept. of ISE)

Department of Information Science & Engineering

Malnad College of Engineering

Hassan– 573202

2025-2026

Index

1. Abstract

The Travel Booking System is a fully functional web-based application built using Django, designed to modernize and simplify the entire process of planning and booking travel services. It provides users with a convenient platform to browse various destinations, explore detailed travel packages, check hotel options, compare pricing, view photos, and place bookings seamlessly from anywhere. The system integrates features like user registration, login, secure authentication, booking history tracking, and real-time availability checking. On the administrative side, it offers a complete management dashboard for handling destinations, hotels, packages, user accounts, payments, cancellations, and booking updates, ensuring smooth backend operations. By eliminating traditional manual paperwork and replacing it with an automated digital solution, the system reduces human errors, saves time, improves accuracy, enhances transparency, and provides a far more efficient travel planning experience. Overall, it delivers a reliable, user-friendly, and well-organized platform that benefits both customers and administrators by making travel booking faster, smarter, and highly convenient.

Developed using Django, HTML, CSS, and SQLite, the system follows best software engineering practices including CSRF security, modular development, role-based user access, and data persistence through Django ORM. These technologies contribute to a scalable and secure architecture, capable of expansion with advanced features such as data visualization, budgeting insights, and automated reminders in future iterations. Overall, the Expense Tracker System serves as a reliable and user-friendly financial management tool that empowers users to take control of their spending patterns, develop financial discipline, and make better economic decisions in their daily lives.

The structured dashboards and clean interface support clarity and improve daily usability, reducing friction that commonly prevents individuals from maintaining financial records over long periods. The system's focus on practical interaction and intuitive workflow ensures that even non-technical users can navigate and operate it comfortably

2. Introduction

Background of the Problem

Travel planning traditionally involves contacting travel agencies, searching multiple websites for hotels and transport, and manually comparing prices. This process is time-consuming, confusing, and often leads to errors in booking. Users do not have a centralized platform where all travel-related information—destinations, packages, hotels, and pricing—can be viewed and booked in one place. As travel demand increases, manual systems fail to provide quick access, real-time availability, or efficient management of bookings.

Why the Domain Was Chosen

The travel and tourism domain is one of the fastest-growing industries, with a significant shift toward online booking platforms. Most people now prefer to book trips digitally due to convenience, transparency, and efficiency. Choosing this domain allows us to solve a real-world problem and build a practical application that can be used by both travelers and administrators. It also provides an opportunity to apply full-stack development skills to create a meaningful, user-friendly web system.

Real-World Scenario Description

In a real travel scenario, a user planning a trip needs to search for destinations, compare packages, find suitable hotels, and check availability. Many people rely on travel agents or scattered online information, which can be unreliable or outdated. A centralized travel booking system helps users explore destinations, check package prices, book services instantly, and receive confirmation—all from a single platform. Similarly, travel companies require a system to manage packages, hotels, users, and bookings without paperwork.

Example1:
A person wants to book a trip to Goa, but they must check hotels on one website, check transport on another, and call a travel agent for package details. This becomes confusing and slow.

Example2:
A family planning a vacation to Ooty has to search manually for hotels, compare prices, and confirm availability. Without a single platform, they waste a lot of time.

Issues in Existing System

- Manual booking is slow and prone to human error.
- Users need to visit multiple websites to compare travel options.
- No centralized system for package details, pricing, and availability.
- Lack of real-time updates on bookings.
- Difficult for admin to maintain large amounts of data manually.
- No booking history or digital record for the user.
- Inconsistent communication and delays in confirmation.

How the Proposed Solution Helps

- Provides a unified online platform for viewing and booking travel packages.
- Users can easily explore destinations, check details, and book instantly.
- Admin can manage hotels, packages, and bookings in one system.
- Reduces manual errors by using automated data handling.
- Offers real-time availability and instant confirmation.
- Enhances user experience with an intuitive and visually appealing interface.
- Stores all booking records digitally, making retrieval quick and easy.
- Improves efficiency, accuracy, and reliability of the entire travel process.

3. Objectives of the Project

The main objectives of the Travel Booking System are as follows:

- To build an online platform for managing travel packages, hotel details, destinations, and user bookings.
- To simplify the process of finding destinations, comparing prices, and booking travel services.
- To provide users with a convenient and user-friendly interface for browsing and booking travel packages.
- To automate the traditional manual processes of travel booking, hotel reservations, and inquiry handling into a fully digital system.
- To offer secure access to user accounts, booking history, admin features, and database operations.
- To improve efficiency and reduce errors in travel planning, booking confirmation, and data management.
- To enable real-time access to available packages, pricing, and hotel availability.
- To centralize all travel-related information such as destinations, hotels, transport, and package details into a single platform.
- To provide admin with easy tools to add, edit, update, and delete travel packages and manage user bookings.
- To ensure accurate and reliable storage of data using a robust backend database.
- To support faster decision-making by allowing users to compare options quickly.
- To create a scalable system capable of handling more destinations, features, and users in the future.
- To offer instant booking confirmation without delays.
- To reduce dependency on travel agents by allowing direct access to travel information.
- To maintain a digital record of all user bookings for quick and easy retrieval.
- To enhance the overall travel experience by providing a smooth, well-structured, and organized booking workflow.

4. System Requirements

4.1 Software Requirements

Programming Languages & Frameworks

- Python 3.10+ (or any Python 3.x version used in your project)
- Django 4.x / 5.x (mention the exact version you used)
- HTML5 for creating the structure of the web pages
- CSS3 for styling the website
- Bootstrap 4/5 for responsive design
- JavaScript for dynamic client-side interactions

Development Tools

- **IDE / Code Editor:**
  - Visual Studio Code
  - PyCharm
  - Sublime Text *(any one you used)*
- **Browser for Testing:**
  - Google Chrome / Firefox / Edge

Database Requirements

- **SQLite3** (default Django database)
  or
- **MySQL** (if used for production setup)
- Django ORM for database operations

Backend Dependencies & Packages

- **Django Admin** for managing backend data
- **Django Authentication System** for login & user access
- **Pillow** (if you are uploading images for packages)

- **Django Crispy Forms** (optional, if used for clean forms)
- **REST Framework** (only if API features are included)

Server Requirements

- **Django Development Server** (python manage.py runserver)

Design & Documentation Tools

- **Canva** for UI design and mockups
- **MS Word / Google Docs** for report writing
- **Draw.io / Figma** for architecture diagrams and ER diagrams

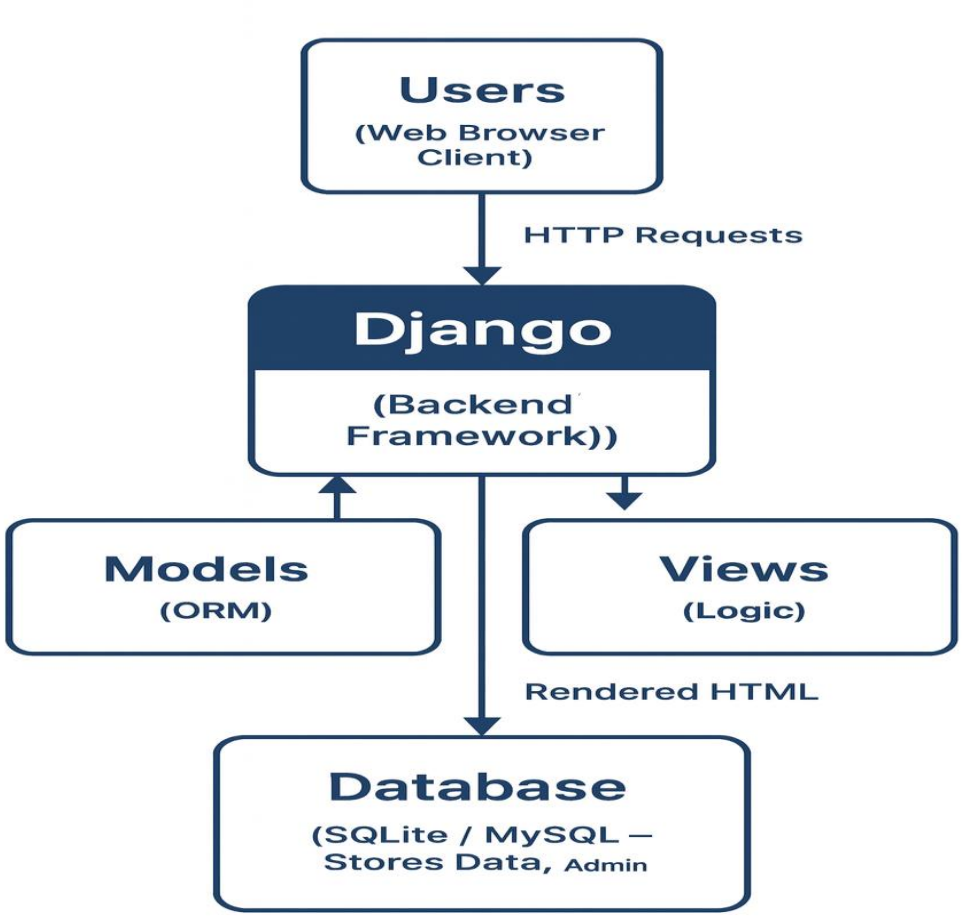
4.2 Hardware Requirements

The minimum hardware configuration required to develop and run the Travel Booking System is:

- Processor: Dual-Core processor (Intel i3 or equivalent)
- RAM: Minimum 4 GB (8 GB recommended for faster performance during development)
- Hard Disk: At least 500 MB free space for project files, Python environment, and dependencies
- Monitor: Minimum resolution 1024 × 768
- Keyboard & Mouse: Standard input devices
- Operating System: Windows / Linux / macOS capable of running Python & Django
- Internet Connection: Required for installing Python packages, Django, and for testing

5.System Design

5.1 Architecture



- **Users** send requests from browser →
- Django **Views** process the request →
- Views interact with **Models** (database tables) →
- Data is returned to **Templates** for display →
- Final HTML page goes back to **User**.

5.2 Explanation of Architecture

1 Users (Client Side)

This is the front-end interface where users interact with the system. They use a web browser to view pages, search destinations, check packages, and make bookings.

2 Django Framework (Backend)

Django handles all the backend logic of the application. It processes user requests, interacts with the database, applies business rules, and returns results.

3. Views

Views contain the logic for handling user requests. When a user searches for a package or tries to log in, the view processes this request and decides what data to fetch or what page to show.

4 Models

Models represent the database structure. They define tables such as User, Package, Hotel, and Booking. Django ORM converts Python objects into database queries.

5 Templates

Templates contain HTML, CSS, and dynamic content. They display information to the user, such as available packages, login form, booking details, or confirmation messages.

6 Database (SQLite/MySQL)

The database stores all persistent information such as user accounts, hotel data, travel packages, booking details, and admin records.

5.3. Data Flow Explanation

- The user performs an action (e.g., search, login, book a trip).
- The request is sent to the Django backend through URLs.
- Django routes the request to the appropriate View function.
- The view interacts with the Model to access or store data.
- The model communicates with the Database using Django ORM.
- The retrieved data is sent back to the View.
- The view renders a Template with the data.
- The final HTML response is displayed to the user.

5.4. User Request Flow (Models → Views → Templates)

Step 1: User Request

The user clicks a page or performs an action such as searching a package, logging in, or booking a hotel.

Step 2: View Processing

Django receives the URL request and maps it to a specific view.
The view processes logic such as verifying user input, checking availability, or fetching hotel/package details.

Step 3: Model Interaction

If data is needed, the view communicates with the model.
The model interacts with the database using Django ORM to:

- Retrieve details
- Store booking data
- Validate user login
- Update records

Step 4: Template Rendering

The view selects an appropriate template and passes the data to it.
Templates display results such as:

- Available packages
- Booking confirmation
- User dashboard
- Admin details

Step 5: Response to User

The final rendered HTML page is sent back to the user's browser.

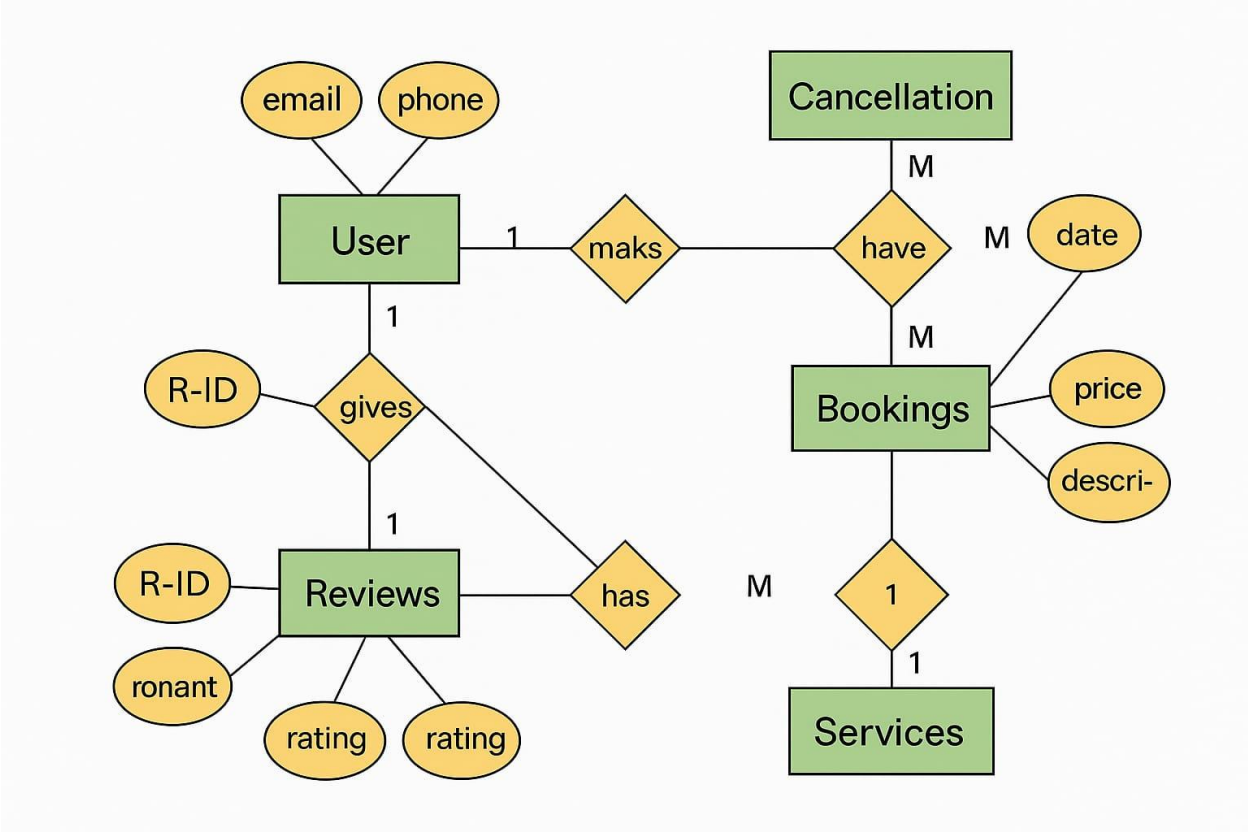
5.4. Interaction with Database

- Django Models use ORM (Object Relational Mapping) to communicate with the database.
- No SQL queries are written manually; Django converts Python code into SQL queries.
- The database stores:
    - User details
    - Hotel information
    - Travel packages
    - Booking history
    - Admin information
- Common interactions include:
    - Insert: When a user books a package
    - Retrieve: When a user views available packages
    - Update: Admin updates package details
    - Delete: Removing old or expired travel packages

The interaction is fast, secure, and efficient.

6. Database Design

6.1 ER Diagram / Schema Diagram



6.2 Tables with Attributes

1. User Table

Attribute	Description
user_id (Primary Key)	Unique ID for each user
name	Full name of the user
email	User login email
password	Encrypted password

phone	Contact number

2. Package Table

Attribute	**Description**
package_id (Primary Key)	Unique ID for each travel package
destination	Travel destination name
description	Details about the package
price	Package cost
days	Number of days in the travel package
image	Image of the destination/package

3. Hotel Table

Attribute	**Description**
hotel_id (Primary Key)	Unique ID for each hotel
name	Hotel name
location	City or destination
price	Price per night
rating	Hotel rating (1–5 stars)

4. Booking Table

Attribute	Description
booking_id (Primary Key)	Unique booking ID
user_id (Foreign Key)	ID of the user who booked
package_id (Foreign Key)	ID of the booked package
booking_date	Date when booking was made
status	Booking status (Confirmed / Pending / Cancelled)

6.3 Keys Used

Primary Keys (PK):

- user_id → User table
- package_id → Package table
- hotel_id → Hotel table
- booking_id → Booking table

Foreign Keys (FK):

- user_id in Booking → references User(user_id)
- package_id in Booking → references Package(package_id)

(Foreign keys ensure relationships between tables.)

6.4 Explanation for Each Table

User Table

This table stores all registered user information such as name, email, and phone number. It is used for authentication and to link each booking with a specific user.

Package Table

Holds details of all available travel packages including destination, price, duration, and description. Users view these packages to decide their travel plan.

Hotel Table

Stores information about hotels available in different destinations. This includes location, room cost, and ratings. It helps users choose the best accommodation.

Booking Table

This table records all booking transactions made by users. Each booking references a user and a package. It stores booking date and current status to manage booking history.

7. Implementation

7.1 Models Overview

The system uses Django models to represent the main entities in the travel booking domain.

- User Model:
  Stores user information such as name, email, phone number, and password. It is used for authentication and linking bookings to specific users.
- Package Model:
  Represents travel packages available in the system. Includes details like destination, description, price, duration (days), and image.
- Hotel Model:
  Contains hotel-related information such as hotel name, location, rating, and cost per night. It is linked to destinations or packages.
- Booking Model:
  Stores details of user bookings including the user ID, package ID, booking date, and booking status (confirmed/pending/cancelled).

These models interact via Django ORM to perform CRUD operations on the database.

7.2 URL Routing Overview

The system uses Django's urls.py to map URLs to specific views. Each feature of the system is connected to a unique URL pattern.

Examples of routing structure:

- URLs for user-related pages (login, logout, register)
- URLs for displaying packages and hotel details
- URLs for booking functionality
- Admin-related URLs for adding/updating packages or hotels
- Homepage, dashboard, and booking history routes

The routing mechanism ensures smooth navigation across all modules of the application.

7.3 Important Functionalities

1. User Authentication

- User registration with validation
- Login and logout using Django's authentication system
- Secure session-based access

2. Package Browsing

- Users can view all available travel packages
- Each package displays details such as destination, price, image, and days included
- Option to view complete package details

3. Hotel Management

- Admin can add, update, or delete hotels
- Users can view hotels included in each package (if linked)

4. Booking System

- Users can book travel packages online
- Booking date and status are recorded in the database
- Users can view their booking history
- Admin can track all bookings

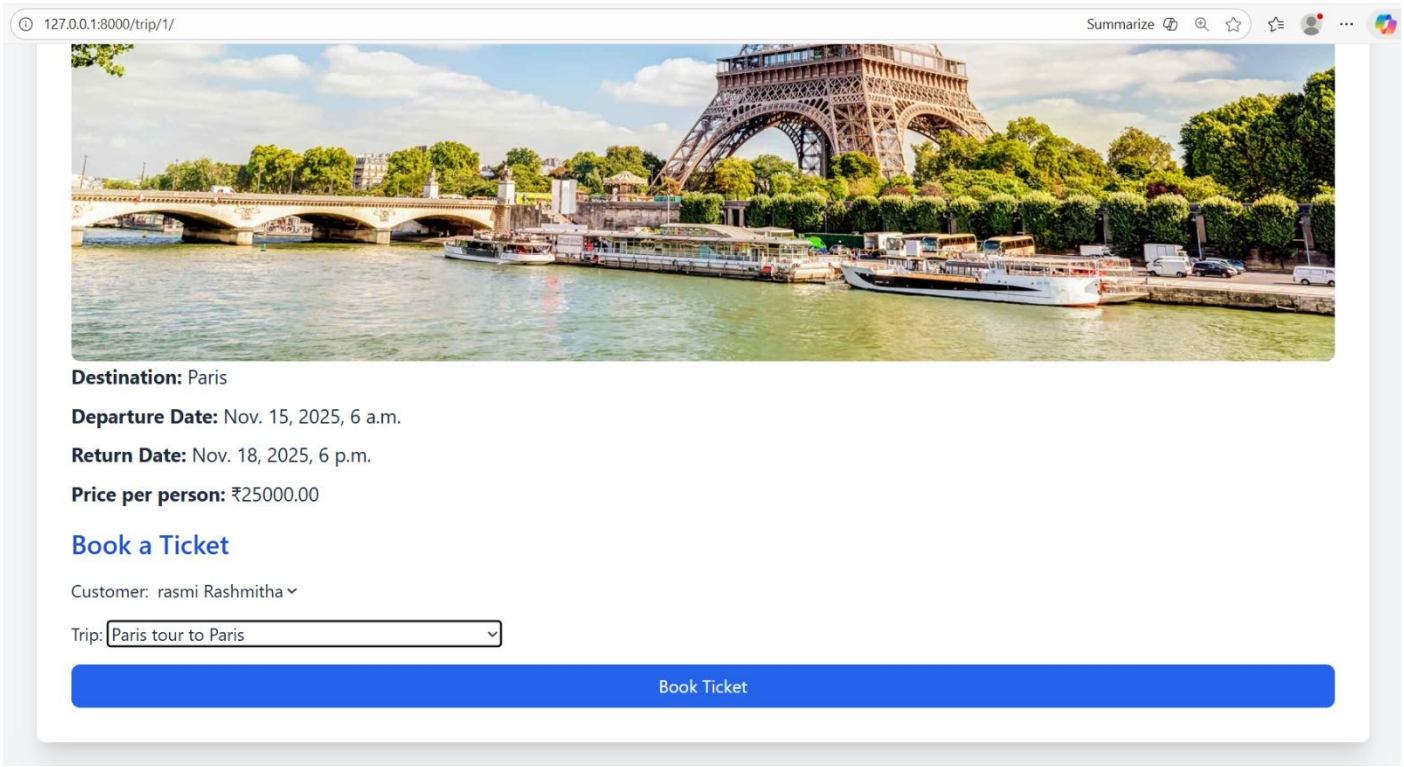
5. Admin Dashboard

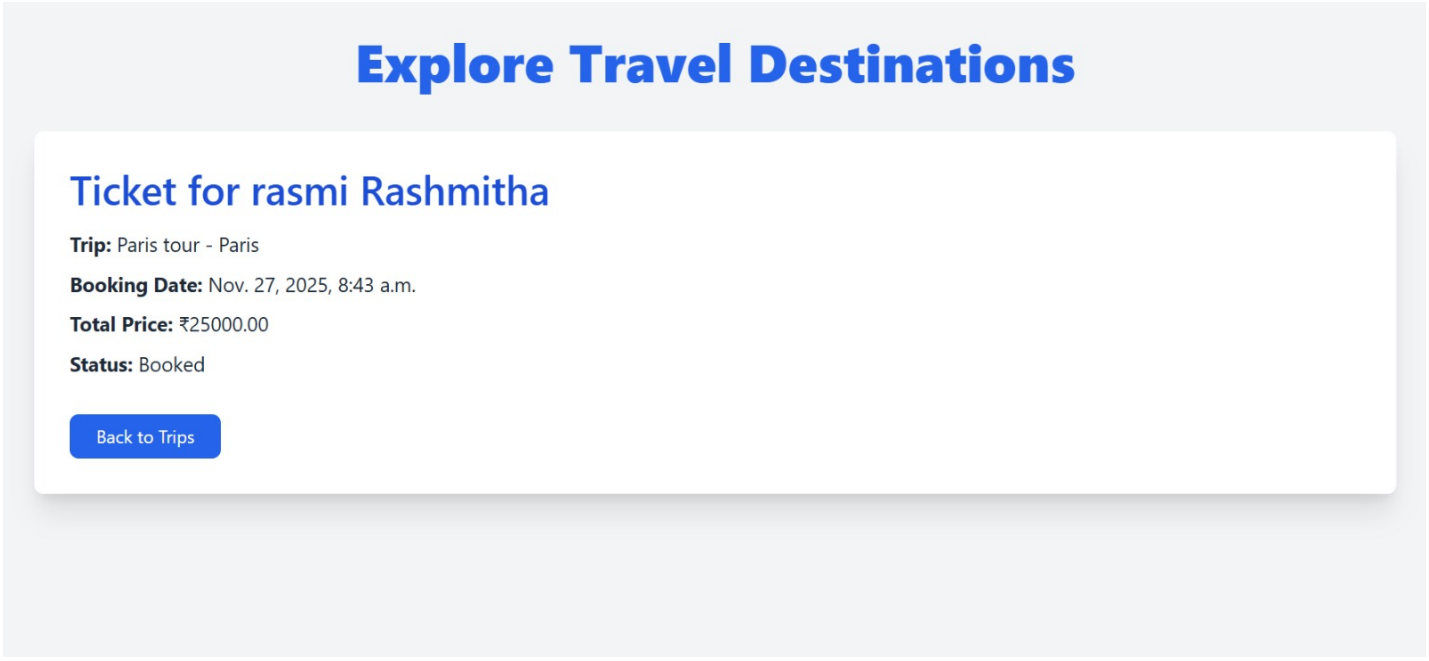
- Manage packages, hotels, and bookings
- View user list and booking records
- Efficient management through Django Admin panel

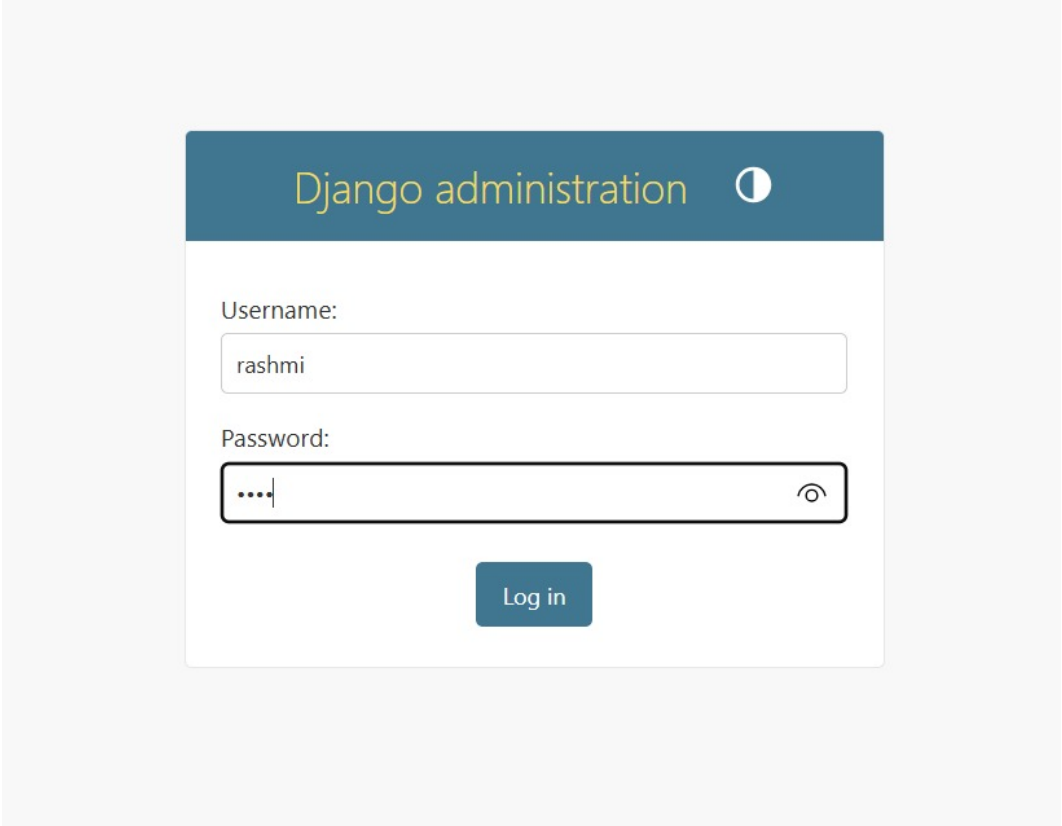
7.4 Special Logic

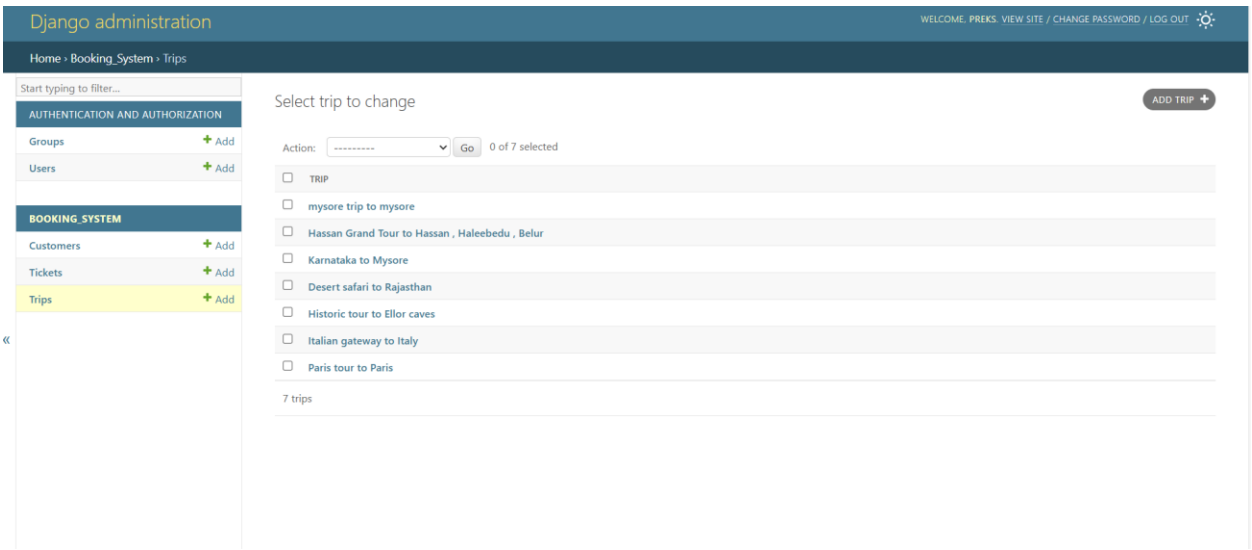
- Dynamic Price Calculation:
  Price may be calculated or adjusted dynamically based on number of days or selected package.
- Login Protection:
  Only authenticated users can access booking pages and booking history. Unauthorized access is blocked using Django decorators.
- Image Handling:
  Package and hotel images are uploaded and managed using Django's media file handling system.
- Automatic Booking Status:
  On successful booking, the system automatically sets the booking status to "Confirmed".
- User-Specific Dashboard:
  Dashboard pages show content specific to the logged-in user only.

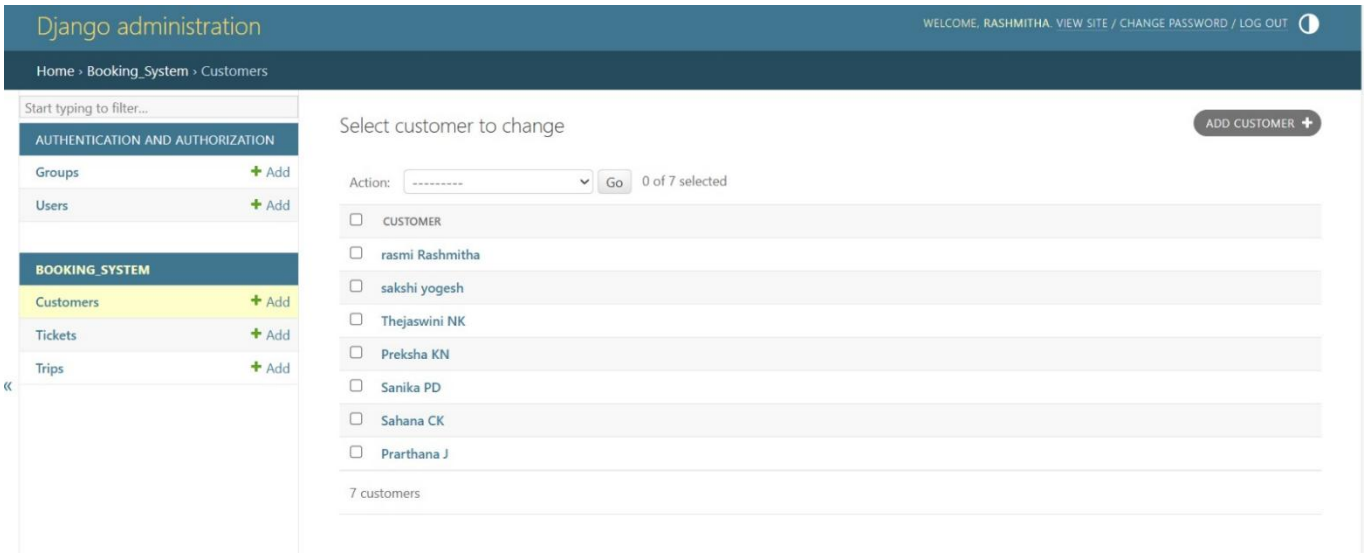
8.Screenshots

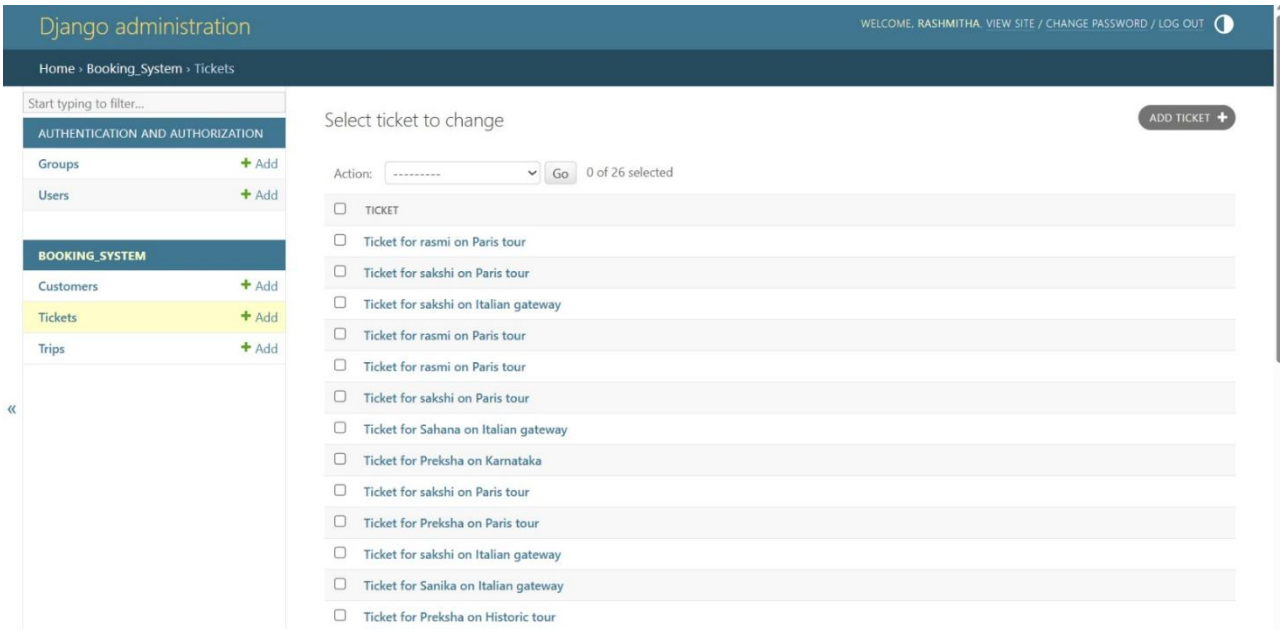





Django administration
Username:
rashmi
Password:
Log in


Django administration
WELCOME, PREKS. VIEW SITE / CHANGE PASSWORD / LOG OUT
Home › Booking_System › Trips
Start typing to filter...
AUTHENTICATION AND AUTHORIZATION
Groups + Add
Users + Add
BOOKING_SYSTEM
Customers + Add
Tickets + Add
Trips + Add
Select trip to change
ADD TRIP +
Action: --------- Go 0 of 7 selected
TRIP
mysore trip to mysore
Hassan Grand Tour to Hassan , Haleebedu , Belur
Karnataka to Mysore
Desert safari to Rajasthan
Historic tour to Ellor caves
Italian gateway to Italy
Paris tour to Paris
7 trips


Django administration
WELCOME, RASHMITHA. VIEW SITE / CHANGE PASSWORD / LOG OUT
Home › Booking_System › Customers
Start typing to filter...
AUTHENTICATION AND AUTHORIZATION
Groups + Add
Users + Add
BOOKING_SYSTEM
Customers + Add
Tickets + Add
Trips + Add
Select customer to change
ADD CUSTOMER +
Action: --------- Go 0 of 7 selected
CUSTOMER
rasmi Rashmitha
sakshi yogesh
Thejaswini NK
Preksha KN
Sanika PD
Sahana CK
Prarthana J
7 customers


Django administration
WELCOME, RASHMITHA. VIEW SITE / CHANGE PASSWORD / LOG OUT
Home › Booking_System › Tickets
Start typing to filter...
AUTHENTICATION AND AUTHORIZATION
Groups + Add
Users + Add
BOOKING_SYSTEM
Customers + Add
Tickets + Add
Trips + Add
Select ticket to change
ADD TICKET +
Action: --------- Go 0 of 26 selected
TICKET
Ticket for rasmi on Paris tour
Ticket for sakshi on Paris tour
Ticket for sakshi on Italian gateway
Ticket for rasmi on Paris tour
Ticket for rasmi on Paris tour
Ticket for sakshi on Paris tour
Ticket for Sahana on Italian gateway
Ticket for Preksha on Karnataka
Ticket for sakshi on Paris tour
Ticket for Preksha on Paris tour
Ticket for sakshi on Italian gateway
Ticket for Sanika on Italian gateway
Ticket for Preksha on Historic tour

9. Testing

Testing was performed to ensure the Travel Booking System works correctly, handles errors properly, and provides a smooth user experience. Various functional and validation tests were carried out on the system. The main test cases are listed below.

9.1 Form Validation Testing

Test Case	Expected Result	Actual Result	Status
User registration with valid details	Account should be created	Account created successfully	Passed
Registration with missing fields	Show error message	Error message displayed	Passed
Booking form with empty fields	Should not allow submission	Validation error shown	Passed
Login form with invalid email format	Should show validation error	Error displayed	Passed

9.2 Login Testing

Test Case	Expected Result	Actual Result	Status
Login with correct username/password	Redirect to homepage	Successfully redirected	Passed
Login with incorrect password	Show “Invalid Credentials”	Error message displayed	Passed
Login without entering any data	Validation errorppee\\bb66]]00	Correct message shown	Passed

Accessing protected pages without login	Redirect to login page	Correct redirection	Passed

9.3 CRUD Operation Testing (Admin Module)

Action	Expected Output	Actual Output	Status
Add new travel package	Package should be saved and visible	Successfully saved	Passed
Edit/update package	Updated details should appear	Changes visible	Passed
Delete package	Package should be removed	Deleted successfully	Passed
Add/edit/delete hotels	Database should reflect changes	Functioning correctly	Passed

9.4 Error Handling

Scenario	Expected Outcome	Actual Outcome	Status
Wrong URL entered	Show 404 error page	404 page displayed	Passed
Server error	Show “Something went wrong” page	Error handled gracefully	Passed
Accessing admin pages without permission	Should show “Access Denied”	Correct restriction applied	Passed
Invalid package or booking ID	Show error message or redirect	Properly redirected	Passed

10. Results

The Travel Booking System successfully achieves its primary goal of providing a centralized, user-friendly online platform for planning and booking travel packages. The system allows users to easily browse destinations, view detailed package information, check hotel details, and make bookings without the need for manual communication with travel agents.

The project effectively solves the major problems in the travel domain by automating traditional booking processes. It eliminates issues such as manual errors, delays in confirmation, scattered information, and lack of real-time availability. With a structured database and Django backend, the system ensures accurate storage and retrieval of user, package, and booking data.

The results demonstrate that:

- Users can book travel packages quickly and securely.
- Admins can efficiently manage packages, hotels, and bookings through the admin panel.
- Form validation, login authentication, and error handling functions work smoothly.
- All important operations (searching, viewing, booking, managing) are performed accurately.
- The entire booking flow—from browsing to confirmation—is completed in a seamless manner.

Overall, the system meets the project objectives, improves efficiency, reduces errors, and provides a simplified and modern approach to travel booking.

11. Conclusion

The Travel Booking System provides a convenient digital solution for users to explore destinations, view travel packages, and make bookings easily. By replacing manual booking methods with an online platform, the system improves efficiency, reduces errors, and offers quick access to important travel information. Users benefit from a smooth interface, instant confirmations, and secure access to their booking history, while admins can manage packages and hotels more effectively.

Overall, the project successfully meets its objectives by creating a user-friendly, organized, and reliable travel booking platform. It demonstrates the usefulness of web-based systems in simplifying real-world processes and enhances the overall travel planning experience for both users and administrators.

12. Future Enhancements

The following improvements can be added to enhance the functionality of the Travel Booking System:

1. Integration of Online Payment Gateway
   Enable users to make secure online payments directly through the platform.
2. Real-Time Transport & Hotel API Integration
   Connect with live APIs for flights, buses, trains, and hotels to show real-time availability and prices.
3. Mobile Application Version
   Develop an Android/iOS app for easier access and faster bookings on mobile devices.
4. User Ratings and Reviews
   Allow users to rate packages and hotels, helping others make better travel decisions.
5. Automated Ticket/Invoice Generation
   Generate downloadable PDF tickets or invoices after successful booking.

13.References

- Django Official Documentation https://docs.djangoproject.com
- Python Official Language Documentation https://www.python.org/doc/
- SQLite Database Reference https://www.sqlite.org/docs.html
- HTML and CSS Specifications https://developer.mozilla.org/en-US/docs/Web