

Table Of Contents

[**Introduction**](#_heading=) **8**

[**Definitions, generalizations and abbreviations:**](#_heading=h.vt60nibqkasx) **9**

[**Description of the subject area**](#_heading=h.etb2dtqyj2ds) **11**

[Description of the application process](#_heading=h.fi3t5tvq8jqx) 11

[Functional Requirements](#_heading=h.h0qu1jm34q4b) 11

[Non-functional requirements](#_heading=h.hjxyphlqn22t) 12

[**System Tech Stack**](#_heading=) **13**

[**System Architecture:**](#_heading=h.jo929s3tb7bq) **14**

[**Schema Diagram:**](#_heading=h.91wntwh41isk) **15**

[**Relationships:**](#_heading=h.9pv5gk3dtss9) **16**

[**Software architecture**](#_heading=h.imwb5mimefli) **17**

[1. Frontend:](#_heading=h.ob5hntxamccy) 17

[2. Backend](#_heading=h.9waoz6l9vmi3) 18

[**Development**](#_heading=) **26**

[Implementation of Backend](#_heading=) 26

[**Implementation of the user interface**](#_heading=h.i2bqok58udpp) **27**

[Interface](#_heading=h.gbpndqdyxcay) 27

[Single Hotel Details:](#_heading=h.boqjdecxos4l) 29

[Cart:](#_heading=h.3i6lfq1hf6g) 30

[Payment:](#_heading=h.51bqyhxpsqzn) 31

[**Conclusion**](#_heading=) **32**

# Introduction

The Hotel Booking Web Application is a user-friendly platform developed using Django and Django templates, styled with Bootstrap for an elegant and responsive interface. This system provides a seamless experience for customers to find and book hotels effortlessly. Users can explore various hotels based on location and amenities, view detailed information about each hotel, and select specific room types tailored to their preferences. The application includes an intuitive cart system where customers can add their selected rooms and proceed to a secure payment gateway for completing the booking. Designed with a focus on simplicity and efficiency, this web app streamlines the hotel booking process, making it convenient for customers to plan their stays.

# Definitions, generalizations and abbreviations:

**Browser:** A browser is a specialized program that enables users to access the Internet, search for hotels, view web pages, interact with booking forms, and play multimedia content. The browser acts as a bridge between the user and the web application, ensuring an intuitive and accessible interface for interacting with the Hotel Booking Web App.

**Framework:** Django, a high-level Python web framework, is used to develop the Hotel Booking Web Application. It provides essential tools, libraries, and pre-built structures to simplify the creation of dynamic, secure, and scalable web applications.

**Web App:** This Hotel Booking System is an web based Application which is accessible from any browser and also operate in Mobile as well

**ORM:** Django's built-in Object-Relational Mapping (ORM) is used to handle database interactions for the application. The ORM maps Python objects to database tables, allowing developers to work with database records using Python code instead of raw SQL queries.

**Cart System:** A feature in the Hotel Booking Web App that allows users to select and temporarily store their desired room bookings. The cart system organizes selected bookings and facilitates their transition to payment processing.

**CustomerProfile:** A user-specific entity that stores personal details, preferences, and booking history for each customer. It enhances the user experience by personalizing interactions and streamlining repeat bookings.

**Room:** Represents the various accommodations available within a hotel. Room objects include information such as type, amenities, pricing, and availability, helping users make informed booking decisions.

**Booking:** A confirmed reservation made by a customer for a hotel room. The booking system tracks customer details, check-in and check-out dates, total guests, and payment status.

**Wishlist:** A feature that enables customers to save their favorite hotels for future reference. It enhances user engagement by allowing users to curate their preferred choices easily.

**Review:** Customers can add reviews, add ratings and also add comments to a hotel. Others can view the review and give likes/unlikes

# Description of the subject area

### Description of the application process

The web application being developed is an Hotel Booking Web Application that allows Customer/User to Register and Book their preferred hotel/room on different locations.

### Functional Requirements

1. Customer Registration and Login:
   1. Allow customers to register, log in, and manage their accounts.
2. Hotel Search and Filtering:
   1. Search hotels based on location, rating, amenities, and availability of rooms.
   2. Filter results by room type (single, double, suite), price range, and additional amenities.
3. Hotel Details and Room Information:
   1. Display detailed information for each hotel, including ratings, reviews, location and available rooms.
   2. Show room types, pricing, availability, and features.
4. Room Booking:
   1. Allow customers to select room(s), check availability, and book them for specific dates.
   2. Support multi-room bookings for a single reservation.
5. Payment Processing:
   1. Enable payment options from online using credit cards or debit cards.
   2. Generate booking confirmations after payment is successful.
6. Coupon System:
   1. Provide coupon codes that customers can apply to get discounts on bookings.
7. Customer Booking History:
   1. Let customers view their booking history, including past and upcoming reservations.
   2. Option to cancel or modify bookings before check-in date
8. Admin Management:
   1. Admins can add, update, and delete hotels, rooms, and their details.
   2. Manage room availability, pricing, and types for each hotel.
   3. Create and manage discount coupons with specific rules.
   4. Monitor and manage payments, refunds etc.

### Non-functional requirements

1. Performance:
   1. Ensure fast search and filter performance for large datasets (hotels, rooms).
2. Scalability:
   1. The system should be able to handle a growing number of users, hotels, and bookings over time.
   2. Plan for scaling the server infrastructure if traffic increases significantly.
3. Security:
   1. It secures user authentication and data protection.
   2. Use HTTPS for secure communication and prevent data breaches.
   3. XSS and CSRF related security breaches can be minimized.
4. Usability:
   1. Provide an intuitive user interface (using Bootstrap) that ensures customers can easily navigate, search, and book hotels.
5. Data Integrity:
   1. Maintain data consistency for room availability, bookings, and payments.
   2. Implement proper rollback mechanisms in case of payment failures or booking issues.
6. Availability:
   1. Ensure high availability of the system to handle bookings at all times.
   2. Use backups and redundancies to prevent data loss or downtime.
7. Maintainability:
   1. Keep the code modular and well-documented for easy updates and maintenance.
   2. Ensure the system is easy to manage for admins through a user-friendly admin interface.
8. Database Efficiency:
   1. Use database optimization techniques like indexing, caching, and connection pooling for fast data retrieval.

# System Tech Stack

This Web Application System is designed to work from any Browser. It is an online system so it is easily accessed and updated.

The system is built with Django as the core language. The frontend is made with Bootstrap as Template. The server is hosted & run in Python Anywhere. The Database is made with PostgreSQL.

The Codebase is served in GitHub and it is also associated in Python Anywhere.

# System Architecture:

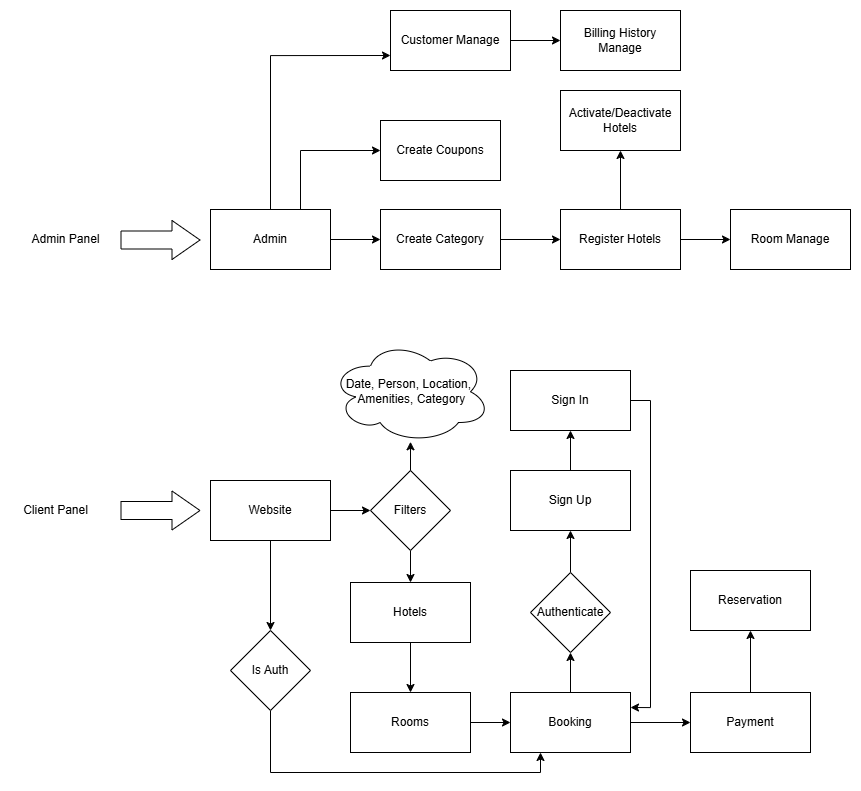


Figure 1. System architecture of the application

# Schema Diagram:

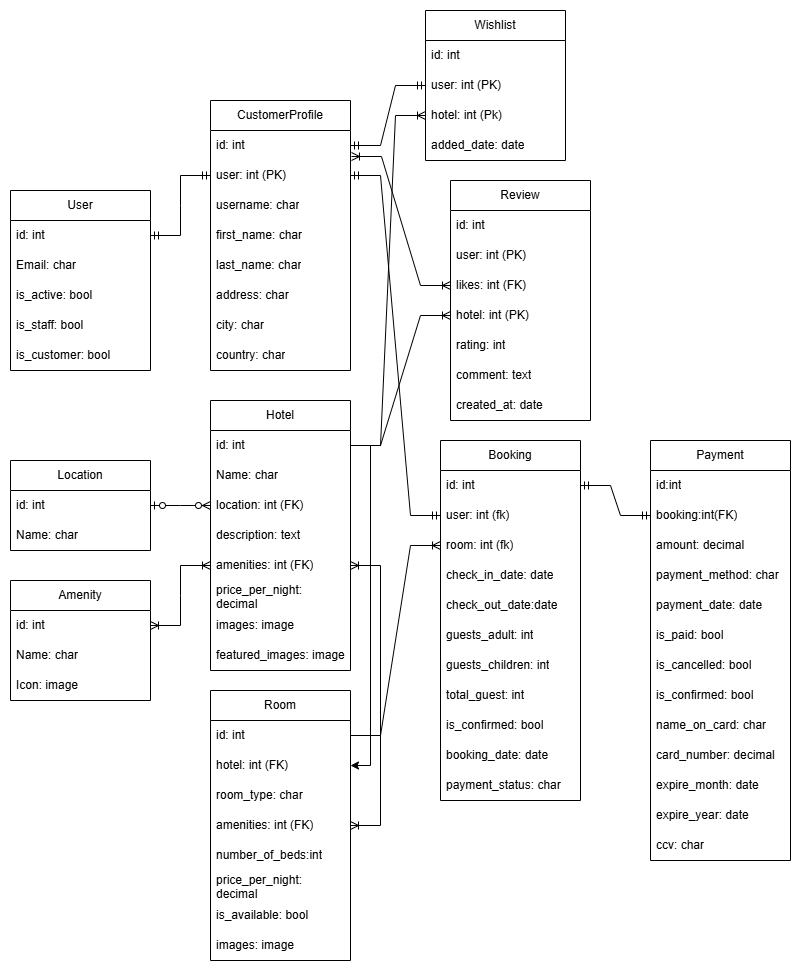


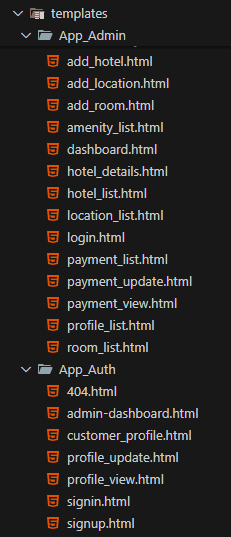
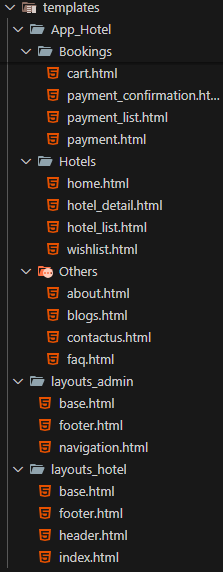
Figure 2. Schema of database tables

# Relationships:

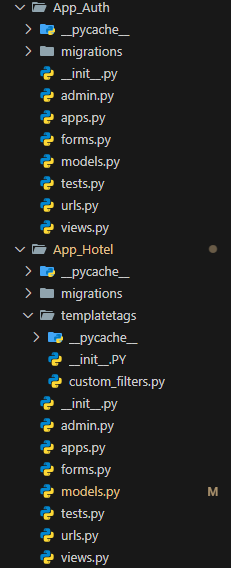
* User - Customer: One-to-One
* Location - Hotel: One-to-Many
* CustomerProfile - Wishlist: One-to-One
* Wishlist - Hotel: One-to-Many
* CustomerProfile - Review: Many-to-Many
* Hotel - Review: One-to-Many
* Amenity - Hotel: Many-to-Many
* Hotel - Room: One-to-Many
* Amenity - Room: Many-to-Many
* Room - Booking: One-to-Many
* CustomerProfile - Booking: One-to-One
* Booking - Payment: One-to-One

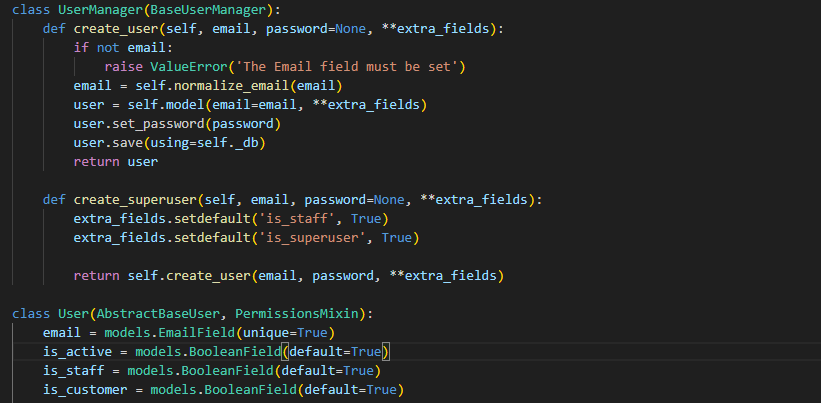
# Software architecture

### Frontend:



### Backend



**App\_Auth:**  
  


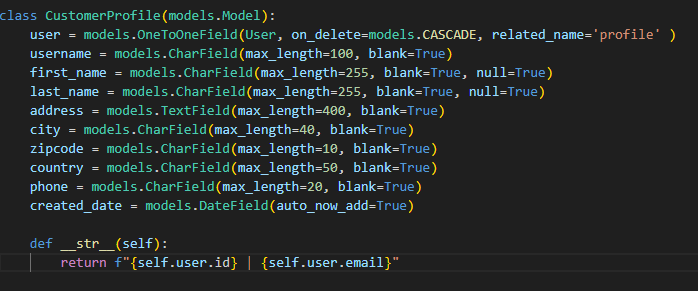
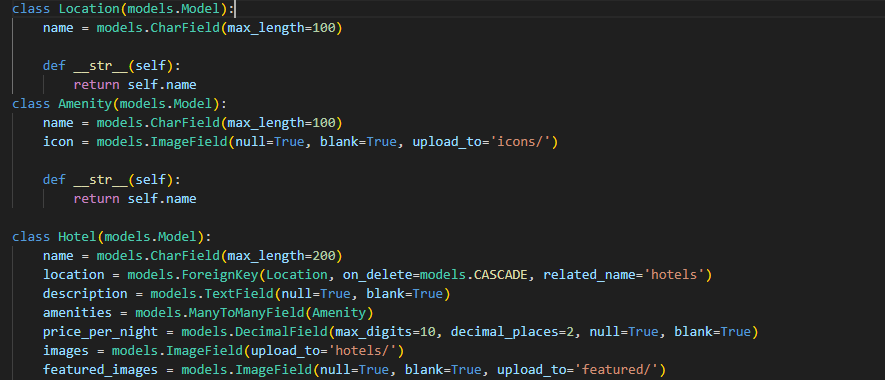


Table 1. Relations of modules and classes

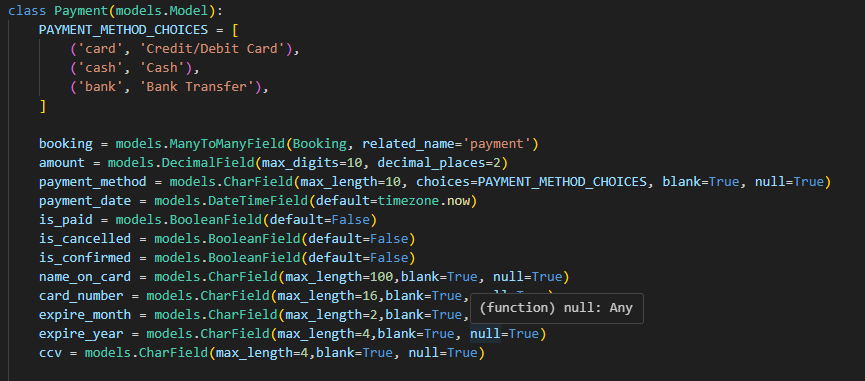
| **Module name** | **Class name** | **Class Assignment** |
| --- | --- | --- |
| App\_AuthModule | User | Model |
| CustomerProfile | Model |
| Create\_User | user creation |
| Create\_SuperUser | superuser creation |

Table 2. Description of classes

| **Class name** | **Description of the class** |
| --- | --- |
| User | Main User Class |
| Create\_User | normal user can be created |
| Create\_SuperUser | superuser/admin user can be created |
| CustomerProfile | Main CustomerProfile |

**App\_Hotel:**  
  




****

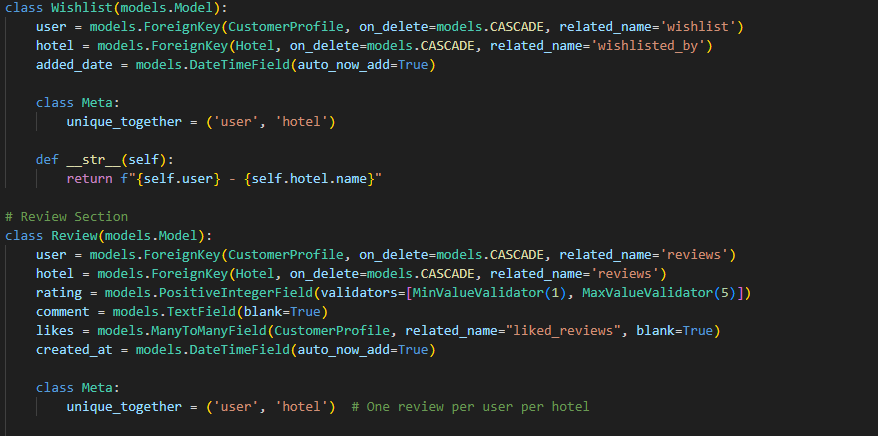
****

Table 3. Relations of modules and classes

| **Module name** | **Class name** | **Class Assignment** |
| --- | --- | --- |
| App\_AuthModule | Location | Represents geographical locations for hotels. |
| Amenity | Represents amenities available for hotels and rooms |
| Hotel | Represents a hotel, linked to a Location and multiple Amenities. |
| Room | Represents rooms in a Hotel, linked to multiple Amenities. |
| Booking | Represents bookings for a Room, linked to a CustomerProfile and related to a Payment. |
| Payment | Represents payment details for one or more Booking instances. |
| Wishlist | Represents a wish list of hotels for a CustomerProfile. |
| Review | Represents reviews for a Hotel, submitted by a CustomerProfile |

Table 4. Description of classes

| **Class name** | **Class description** |
| --- | --- |
| Location | Represents a geographical location that can be associated with one or more hotel. |
| Amenity | Represents a specific amenity, such as free Wi-Fi, swimming pool, or breakfast service. |
| Hotel | Represents a hotel, including its location, amenities, description, and associated images. |
| Room | Represents an individual room in a hotel, with details like type, price, amenities, and availability. |
| Booking | Represents a customer's reservation for a hotel room, including check-in and check-out details. |
| Payment | Represents payment information for one or more bookings, with various payment methods and statuses. |
| Wishlist | Represents a collection of hotels that a customer has added to their wishlist for future reference. |
| Review | Represents a review submitted by a customer for a hotel, including a rating, comments, and likes. |

# Development

## Implementation of Backend

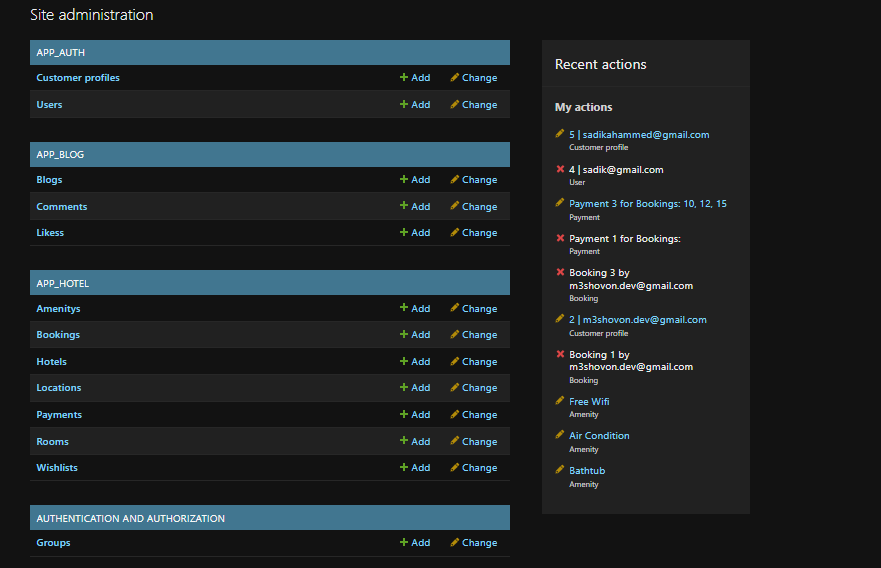
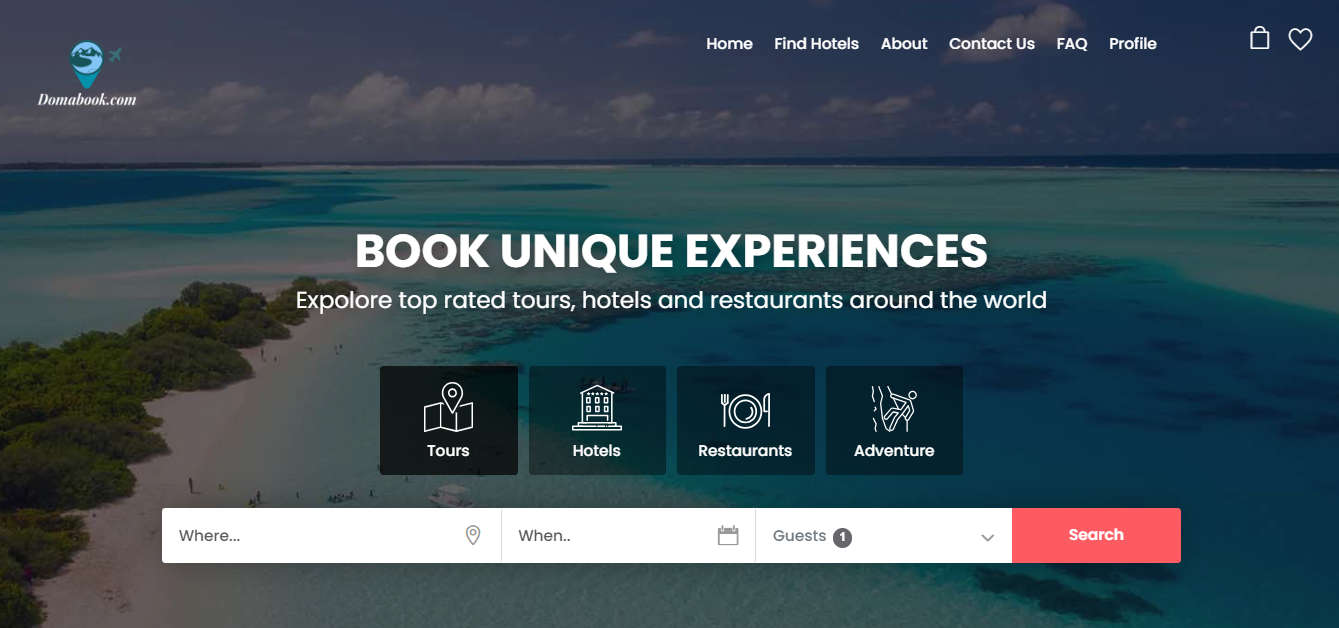
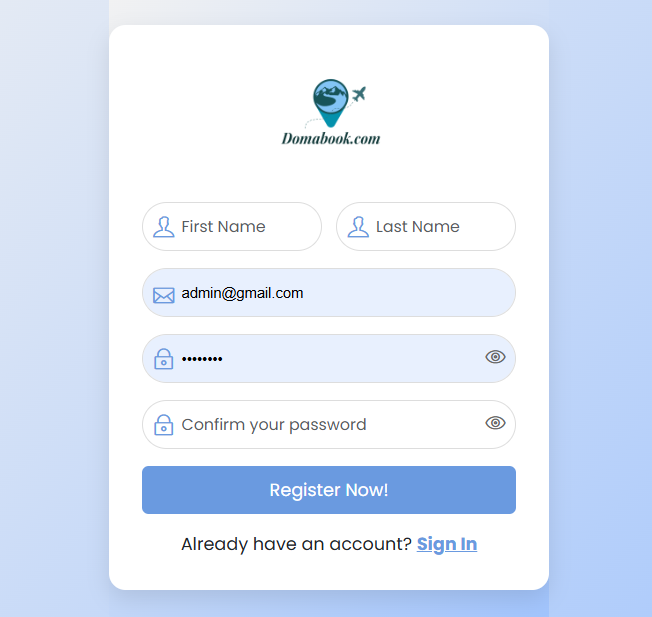


Figure. The Backend Admin panel of the system

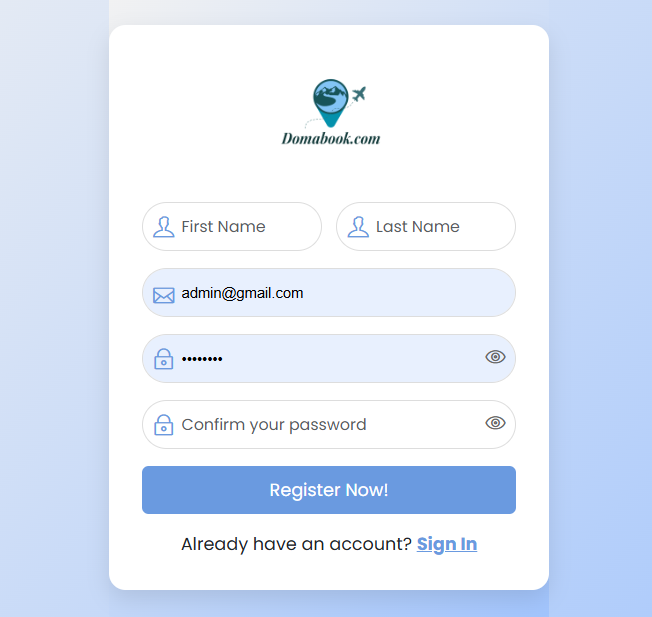
# Implementation of the user interface

### Interface

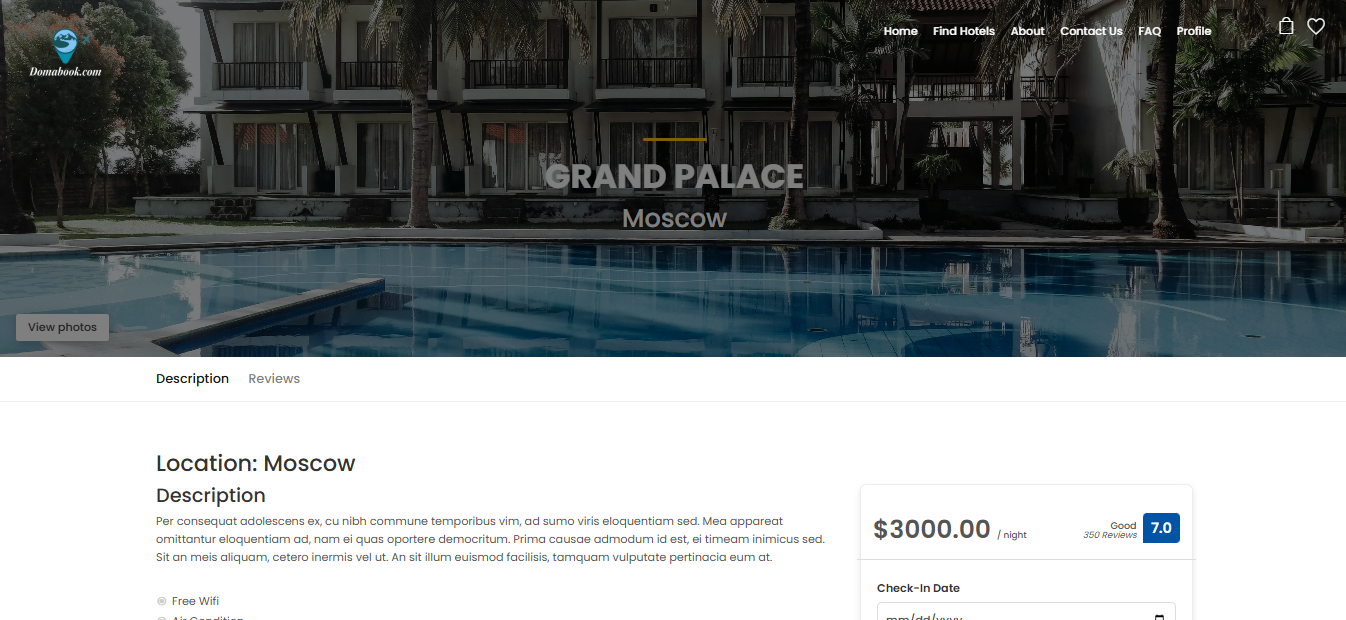


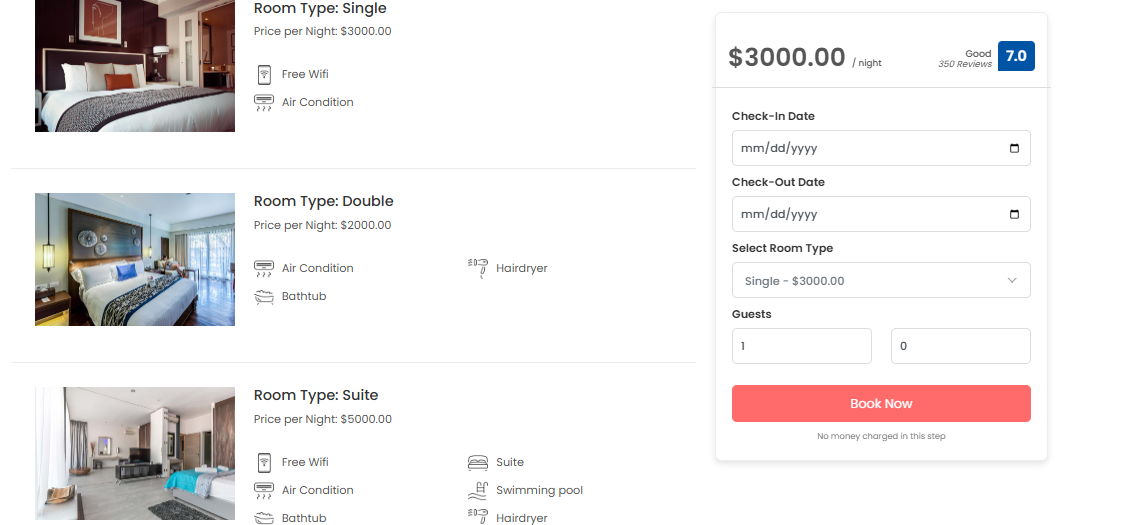
**Signup**

**Signin**



### Single Hotel Details:



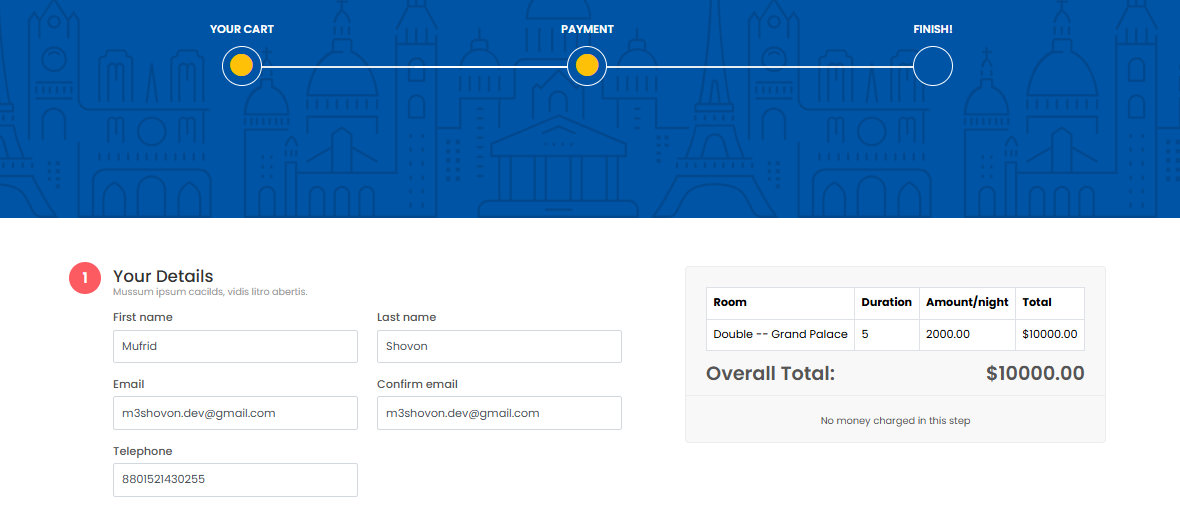


# 

### Cart:

# 

### Payment:



# 

# 

# Conclusion

The Hotel Booking Web Application combines robust backend functionality with a polished frontend to deliver a comprehensive hotel reservation solution. By enabling customers to browse hotels, choose rooms, manage bookings through a cart, and make secure payments, the system ensures a hassle-free booking journey. With its scalable design and responsive interface, the application caters to diverse user needs, making it an ideal platform for modern travelers. This project demonstrates the potential of Django and Bootstrap in creating efficient, dynamic, and user-centered web applications.