Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

StayEase: A Hotel Booking Management System

Chirag Agarwal, Gaurav Kothi, Himanshu Jangid, Kavya Joshi, Nidhi Gour
Department of Computer Science and Engineering, JECRC University, Jaipur, India

Abstract - The hospitality industry has rapidly shifted toward digital reservation platforms; however, many small and medium-sized hotels still rely on manual or semi-digital processes due to the high cost and complexity of commercial online travel agency (OTA) solutions. This paper presents StayEase, a web-based hotel booking and management system designed to automate reservation workflows while remaining affordable, customizable, and easy to deploy. StayEase enables travelers to search and book rooms online and provides hotel administrators with a centralized dashboard to manage room inventory, bookings, and reports. The system is implemented using HTML, CSS, and JavaScript for the frontend, Django (Python) for the backend, and PostgreSQL for data storage, following a three-tier architectural model to ensure modularity, security, and scalability. Performance testing under simulated load conditions demonstrated average response times below 3.2 seconds for up to 200 concurrent users. User acceptance testing with 20 participants reported an overall satisfaction rate of 96%. These results indicate that StayEase effectively streamlines hotel reservation operations and offers a practical alternative to commission-based OTA platforms for smaller hospitality providers.

Keywords: Hotel Booking System, Web Application, Django, Software Engineering, Hospitality Management

INTRODUCTION

The hospitality sector has undergone a significant digital transformation over the past two decades. Traditional hotel reservation methods based on phone calls, physical registers, and walk-in customers have increasingly been replaced by online booking platforms that offer real-time availability, instant confirmation, and improved convenience. With widespread internet access and the growth of mobile devices, travelers now expect seamless digital experiences when planning accommodation.

Large online travel agencies (OTAs) such as Booking.com and Expedia have contributed greatly to this transformation by aggregating hotel listings and enabling global reservations. However, these platforms often impose high commission fees and offer limited customization, which can negatively affect the profitability and operational autonomy of small and independent hotels. Additionally, dependence on third-party platforms can limit direct customer engagement.

To address these challenges, StayEase was developed as a self-hosted, web-based hotel booking management system. The objective of StayEase is to provide hotels with a simple yet robust platform that automates reservation processes while allowing full control over system configuration and data. By leveraging modern web technologies and established software engineering practices, StayEase aims to improve operational efficiency for hotel staff and enhance booking convenience for guests.

LITERATURE REVIEW

Online hotel reservation systems have evolved significantly since their early adoption in the late 1990s. Initial solutions were often standalone desktop applications with limited functionality and no real-time connectivity. The advancement of web technologies enabled the development of online platforms that allow customers to search availability and confirm bookings remotely.

Studies in hospitality information systems indicate that digital reservation platforms reduce operational overhead and minimize human errors compared to manual processes. However, existing literature also highlights the disadvantages of relying heavily on third-party OTA platforms, such as recurring commission costs, reduced control over customer data, and limited customization for individual hotels.

Recent research emphasizes the importance of modular architecture, secure authentication, and responsive user interfaces in hotel management systems. Frameworks such as Django are frequently recommended due to their built-in security features, scalability, and rapid development capabilities. StayEase incorporates these principles by adopting a three-tier architecture, implementing secure user authentication, and maintaining a normalized relational database to ensure data integrity. Unlike large OTA-based platforms discussed in existing studies, StayEase focuses on a self-hosted and commission-free model tailored for small and independent hotels.

METHODOLOGY

- The development of StayEase followed the traditional Waterfall Software Development Life Cycle (SDLC). This model was selected due to the fixed academic scope of the project, clearly defined requirements, and a limited development timeline, making a sequential approach more suitable than iterative Agile methods.

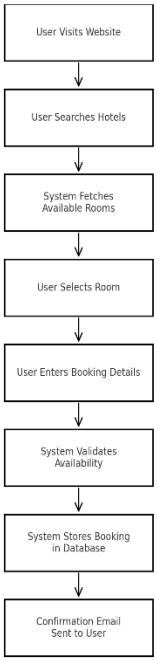
  The key phases of the SDLC included:

- **Requirement Analysis:** Identification of functional requirements such as room search, booking management, and administrative control, along with

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

non-functional requirements related to security, performance, and usability.

- **System Design:** Creation of architectural designs, database schemas, and user interface wireframes. A three-tier architecture was selected to separate presentation, application logic, and data management.
- **Implementation:** Development of backend logic using Django and frontend components using standard web technologies. Database models and business logic were implemented according to the design specifications.
- **Testing:** Execution of unit testing, integration testing, and user acceptance testing to verify system functionality and reliability.
- **Deployment and Maintenance:** Deployment of the application on a local server for demonstration purposes, with plans for future cloud hosting and feature enhancements.



SYSTEM DESIGN

StayEase follows a three-tier architecture consisting of the presentation layer, application layer, and data layer. The presentation layer is implemented using HTML5, CSS3 with Bootstrap, and JavaScript to provide a responsive user interface across devices. The application layer is built using the Django framework, which processes client requests, enforces business logic, and manages authentication through its Model–View–Template (MVT) architecture. The data layer uses a PostgreSQL relational database to store user, hotel, room, and booking information.

The system is organized into the following functional modules:

- User Module: Handles user registration, login, and profile management with secure password storage.
- Hotel Management Module: Allows administrators to add, update, and manage hotel and room details.
- Room Booking Module: Enables users to search for available rooms and make reservations with real-time availability validation.
- Feedback Module: Collects post-stay reviews and ratings from users.
- Reporting Module: Generates administrative reports related to bookings and occupancy.

Database normalization techniques were applied to reduce redundancy and maintain consistency. Relationships between entities are maintained using primary and foreign keys, supporting efficient and reliable data operations.

IMPLEMENTATION

The StayEase system was implemented using Python 3.11 and Django 5.0 for backend development, with PostgreSQL 15 used as the database management system. The frontend was developed using HTML, CSS, JavaScript, and Bootstrap to ensure responsiveness and usability. Development was carried out using tools such as Visual Studio Code, and testing was performed across modern web browsers including Google Chrome and Mozilla Firefox.

Django models were defined for core entities such as users, hotels, rooms, and bookings. Views were implemented to handle HTTP requests, validate user input, and enforce booking constraints such as preventing overlapping reservations. Django's built-in admin interface was customized to support administrative operations. Email notifications were implemented using SMTP to confirm bookings and provide status updates to users.

RESULTS AND TESTING

StayEase was tested extensively to evaluate its performance, reliability, and usability. The testing process included unit testing of Django models and core functions, integration testing of form submissions and backend workflows, and system testing of complete user scenarios such as room search, booking, and cancellation. These tests ensured that individual components and end-to-end processes functioned as intended.

Performance testing was conducted under simulated concurrent user conditions to assess system responsiveness and server load. Table 1 summarizes the observed performance metrics for different usage scenarios.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 12 , December - 2025

Test Case	Avg. Response Time	Server CPU Load	Result
10 users browsing home page	0.7 s	12%	Pass
50 concurrent booking submissions	1.5 s	45%	Pass
100 concurrent users overall	2.3 s	71%	Pass
200 concurrent users overall	3.1 s	88%	Acceptable

Table 1: Sample performance test results for StayEase (under moderate load).

The system demonstrated stable behavior as the number of concurrent users increased. Even with 200 concurrent users, the average response time remained approximately 3.1 seconds with acceptable server resource utilization, indicating that the application can handle moderate real-world traffic efficiently.

User acceptance testing (UAT) was conducted with 20 participants, including 15 potential guests and 5 hotel staff members. Participants performed typical tasks on a live demonstration system and evaluated the application based on usability, booking ease, speed, and interface design. The average ratings ranged between 4.6 and 4.8 out of 5 across all criteria, resulting in an overall user satisfaction score of 96%.

The results indicate that StayEase fulfils its functional objectives by automating the hotel booking process, minimizing manual errors, and centralizing reservation data within a secure database. The system performed reliably across both desktop and mobile browsers. The backend architecture based on Django proved sufficiently scalable for future enhancements such as payment gateway integration and cloud deployment. Current limitations include the absence of a live payment gateway and support for only a single language interface, both of which are planned for future development.

CONCLUSION

This paper presented StayEase, a web-based hotel booking and management system designed to address the operational challenges faced by small and medium-sized hospitality providers. By integrating a responsive frontend with a secure and scalable Django backend, StayEase automates reservation workflows and centralizes data management. The system reduces manual errors, improves booking efficiency, and enhances user convenience.

Future work will focus on integrating a secure online payment gateway, deploying the system on cloud infrastructure, and adding features such as multilingual support and advanced analytics. These enhancements will further strengthen StayEase as a comprehensive digital solution for hospitality management.

Acknowledgements: The authors gratefully acknowledge the guidance of the project supervisor, Ms. Nidhi Gour, and the support provided by JECRC University during the completion of this project.

REFERENCES

[1] Django Software Foundation. (2024). Django: The Web Framework for Perfectionists with Deadlines.
[2] M. Grinberg, Flask Web Development and Django Essentials, O'Reilly Media, 2018.
[3] S. Holzner, Database Programming with Python, McGraw Hill Education, 2020. A. Sharma, Fundamentals of Web Technologies, McGraw Hill Education, 2022.
[4] J. Nielsen, "Usability Heuristics for User Interface Design," Nielsen Norman Group, 2019.
[5] S. Roy, Python Web Development: Concepts and Techniques, PHI Learning, 2021. A. Patel and B. Shah, "Digital Transformation in Hotel Reservation Systems," International Journal of Hospitality Management, vol. 38, no. 2, pp. 85–94, 2022.
[6] R. K. Mishra and P. Sharma, "Security Considerations in Web-Based Applications," Journal of Computer Applications, vol. 14, no. 1, pp. 22–29, 2021.
[7] PostgreSQL Global Development Group, PostgreSQL Documentation, 2024. Available: