

Find My Stay - An Online Hostel Booking System

¹ Athar Fathima

CSE Assistant Professor

Keshav Memorial Institute of Technology Keshav Memorial Institute of Technology Keshav Memorial Institute of Technology

Hyderabad, India

athar.fathima23@gmail.com

² Thotla Ganesh

IV year CSE Student

Hyderabad, India

ganeshthotla594@gmail.com

³ Sugandham Hrishita

IV year CSE Student

Hyderabad, India

sugandham.hrishita@gmail.com

⁴ Meka Keerthika

IV year CSE Student

Keshav Memorial Institute of Technology Keshav Memorial Institute of Technology Keshav Memorial Institute of Technology

Hyderabad, India

keerthikaa0510@gmail.com

⁵ Pendem Sai Madhav

IV year CSE Student

Hyderabad, India

saimadhav204@gmail.com

⁶ Pendem Sanjana

IV year CSE Student

Hyderabad, India

sanjupendem15@gmail.com

Abstract—The Online Hostel Booking System is a web-based application designed to simplify and digitalize the process of searching, comparing, and booking hostel accommodations. It allows users to browse hostels based on key factors such as location, facilities, room type, and budget. Each hostel listing provides detailed information about available rooms, including single, double (2-sharing), triple (3-sharing), and multi-sharing options, along with corresponding prices and amenities. A virtual room preview feature enables users to explore hostel interiors before confirming their booking, improving decision-making convenience. The system utilizes cloud storage to ensure secure and reliable management of user and hostel data. Hostel owners can register their properties, update room details, manage availability, and track bookings through an intuitive admin dashboard. By integrating modern web technologies with cloud computing, the project delivers a transparent, efficient, and user-friendly platform that enhances the overall hostel booking experience for students and travelers.

Index Terms—Hostel management, Artificial Intelligence, LLM, MERN stack, automation, recommendation systems

I. INTRODUCTION

In today's digital world, almost every service has moved online — from shopping to transportation and hotel bookings. However, long-term hostel management still largely depends on manual processes and informal communication. Students and working professionals often struggle to find suitable hostels, check real-time room availability, submit required documents, and complete bookings without confusion or delay. The absence of a structured digital system makes the process time-consuming, unclear, and sometimes frustrating. At the same time, hostel owners face their own set of challenges. Many still rely on physical registers to track room occupancy, manage tenant information, and monitor payments. When managing multiple hostel branches, maintaining accurate and consistent records becomes even more difficult. Manual methods not only increase the risk of human error but also prevent owners from gaining useful insights into vacancy trends, revenue patterns, and tenant demographics. Without a centralized system, decision-making becomes reactive rather than data-driven. To overcome these challenges, this paper introduces a Hostel

Management and Booking System — a centralized web-based platform that simplifies and digitalizes the interaction between tenants and hostel owners. The system allows tenants to search for hostels using filters such as location, price, and room type, upload required identification documents, and complete bookings through a streamlined online process. For hostel owners, the platform provides an organized dashboard that displays real-time occupancy status, tenant records, and financial summaries. By replacing traditional manual processes with a secure and scalable digital solution, the proposed system improves efficiency, transparency, and overall user experience. With features such as role-based authentication, cloud-based document storage, and planned payment gateway integration, the platform is designed to be reliable, future-ready, and adaptable. Ultimately, this system aims to bring structure, clarity, and convenience to long-term hostel management through a unified digital ecosystem.

II. RELATED WORK AND EXISTING SYSTEMS

The digital transformation of the accommodation industry has led to the development of several online booking and property management platforms. Companies such as Airbnb and OYO Rooms have successfully introduced user-friendly systems that allow customers to search properties, compare prices, and complete bookings online. These platforms demonstrate the effectiveness of digital search filters, secure payment integration, and structured booking workflows. In parallel, various academic and institutional hostel management systems have been developed to handle room allocation, tenant record management, and administrative tasks. These systems showcase how digital tools can organize hostel operations and improve data handling efficiency. Additionally, modern booking systems increasingly integrate secure digital payment gateways such as Razorpay, enabling seamless transaction processing and automated confirmation. Building upon these advancements, the proposed Hostel Management and Booking System combines structured booking workflows, role-based access, document handling, and centralized dashboards to cre-

ate a comprehensive and streamlined digital solution tailored for long-term hostel management.

III. DRAWBACKS OF EXISTING SYSTEMS

Traditional hostel booking and management systems still rely heavily on manual methods, which create several practical challenges. Tenants often have to visit multiple hostels in person to check room availability, compare prices, and understand the facilities offered. Since information is not organized in a standardized way, it can lead to confusion, misunderstandings, and delays in making decisions. Document verification is usually handled manually, increasing the risk of misplacement, duplication, or human error. From the hostel owners' perspective, maintaining physical registers for occupancy tracking, tenant records, and payment details can become overwhelming over time. Managing multiple hostel branches further adds to the complexity, making it difficult to maintain consistent and updated records. Without a centralized digital system, owners also lack clear insights into vacancy patterns, revenue tracking, or tenant demographics. Overall, the absence of automation results in inefficiency, limited transparency, and reduced operational control, affecting both tenants and hostel owners.

IV. PROPOSED SYSTEM ARCHITECTURE

The proposed Hostel Management and Booking System follows a scalable multi-tier architecture designed to ensure smooth interaction between tenants, hostel owners, and system services. At the Client Layer, users access the platform through a web browser, where they can search hostels, upload documents, submit booking requests, and view dashboards. All user actions are securely transmitted to the backend. An Nginx reverse proxy manages incoming traffic, improving performance, security, and request handling before forwarding requests to the server. The Backend Layer, built using Node.js and Express, handles core business logic such as booking workflows, document processing, occupancy calculations, and role-based authentication. An integrated analytics service processes hostel data to generate occupancy and financial insights for owners. The Database Layer (PostgreSQL) stores structured data including user profiles, hostel details, room information, and booking records. For secure file storage, AWS S3 and CloudFront are used to manage uploaded documents efficiently. The system also integrates the Razorpay payment gateway to support secure and seamless online transactions. Overall, the architecture ensures reliability, scalability, and secure end-to-end communication within the platform.

V. PROPOSED METHODOLOGY

The development of the Hostel Management and Booking System follows a systematic and user-centered methodology aimed at addressing real-world challenges in hostel operations. The process began with a detailed analysis of existing manual booking and management practices to understand inefficiencies and user pain points faced by both tenants and hostel owners. After identifying the core problems, the system requirements were clearly defined and categorized into functional and

non-functional requirements. Based on these requirements, the system was designed using a modular development approach, separating tenant and owner functionalities to ensure clarity and role-based access control. The implementation phase focused on building core modules step by step. The tenant module was developed to handle hostel search, booking requests, document submission, and payment processing. In parallel, the owner module was designed to manage hostel listings, room configurations, booking approvals, and occupancy monitoring. Business logic was implemented to automate workflows such as request submission, status updates, and data validation. To ensure data consistency and reliability, a structured database design was adopted. Secure authentication mechanisms and document handling procedures were implemented to protect user data. The system was then tested through functional testing, integration testing, and performance validation to ensure smooth interaction between modules. This step-by-step methodology ensures that the proposed system is not only technically sound but also practical, scalable, and aligned with the real needs of hostel management.

VI. ALGORITHM DESIGN

Experimental Setup and Results

This section presents the experimental environment, tools used, and performance evaluation of the proposed cloud-based task management and digital journaling system. The objective of the evaluation is to analyze system performance, security effectiveness, and scalability under realistic usage scenarios.

Experimental Setup

A. System Architecture

The Hostel Booking system is implemented using a three-tier client-server architecture to ensure modularity, scalability, and maintainability. The architecture consists of a backend layer, an owner/admin frontend, and a tenant-facing portal. The backend layer is developed using Node.js and Express 5, providing a RESTful API for handling business logic, authentication, and database interactions. MongoDB is used as the primary database for persistent data storage. The Owner/Admin Frontend is developed using React 19 with Vite 7 as the build tool. TailwindCSS 4 and DaisyUI 5 are used to design a responsive and interactive dashboard for hostel owners and administrators to manage hostels, rooms, applications, and system analytics. The Tenant Portal, also built using React 19 and Vite 7 with TailwindCSS 4, provides a public interface where tenants can browse hostels, compare facilities, submit applications, and track their status. This layered architecture ensures clear separation of concerns and smooth communication between system components.

B. Hardware and Software Environment

The backend server operates on the latest Long-Term Support (LTS) version of Node.js. MongoDB Atlas (cloud-based deployment) is used for database hosting and management. The frontend is built using Vite 7, which supports Hot Module Replacement (HMR) for efficient development. Authentication is implemented using JSON Web Tokens (JWT), with bcrypt used for secure password hashing. Online payment processing is integrated using SSLCommerz, enabling secure rent and fee transactions. Email notifications such as password resets and system alerts are managed using NodeMailer. The system is deployed using Vercel for backend services and Firebase Hosting for the frontend application. Development and testing were conducted on a Windows operating system, and cross-browser compatibility was verified using Google Chrome, Mozilla Firefox, and Microsoft Edge.

C. Database Design

The MongoDB database consists of nine collections designed to manage different system functionalities. The User collection stores authentication credentials and role-based information. The Hostel and Room collections manage hostel metadata, amenities, pricing, and availability. The Application and JoinRequest collections handle tenant onboarding workflows. Additional collections include MealBooking for daily and automated meal reservations, Payment for transaction records via SSLCommerz, Complaint for grievance management, and Notification for system alerts. This structured schema supports efficient querying, scalability, and data consistency.

D. API Design

The backend exposes approximately 40+ RESTful endpoints grouped into 10 route modules. These modules handle authentication (/auth), user management (/users), hostel management (/hostels), room management (/rooms), applications (/applications), join requests (/join-requests), meal bookings (/bookings), payments (/payments), complaints (/complaints), and notifications (/notifications). This modular API structure improves maintainability and ensures organized separation of system functionalities.

E. Security Implementation

Security is enforced through middleware-based access control. JWT-based authentication is validated using a verifyToken middleware. Role-based authorization is implemented using verifyAdmin and verifyAdminOrOwner middleware functions to restrict access to privileged operations. This layered security approach ensures that only authorized users can access protected resources and perform sensitive operations.

VII. RESULTS AND PERFORMANCE ANALYSIS

A. Feature Completeness

The implemented system successfully delivers all core functional modules defined during the requirement analysis phase. Multi-role authentication supports Admin, Owner, and

Tenant roles using secure JWT-based authorization. The tenant portal enables public hostel browsing, search, and side-by-side comparison without mandatory login. Hostel owners can manage hostels and rooms efficiently, including room allocation and inventory updates. The system supports an integrated meal booking module with both daily and automated booking options. Online payments are processed securely through SSLCommerz integration, enabling rent and fee transactions. Additional modules such as complaint management, real-time in-app notifications, and a structured join request workflow enhance user interaction and operational control. Both admin and user dashboards provide visual insights, and email-based password reset functionality ensures account recovery. Overall, the system demonstrates full feature implementation aligned with the project objectives.

B. Frontend Performance

The Owner Dashboard consists of over 43 pages and utilizes more than 18 reusable components, ensuring modularity and maintainability. The Tenant Portal includes four primary pages with shared reusable components for consistency. The frontend is built using TailwindCSS and DaisyUI for responsive design. TanStack React Query is used for efficient state management and API data synchronization, while React Hook Form handles structured form validation. Motion (Framer Motion) enhances user interaction within the dashboard. The application is built using Vite 7, which provides fast Hot Module Replacement (HMR) with refresh times typically under 100 milliseconds during development. This significantly improves development efficiency and frontend responsiveness.

C. Backend Performance

Backend performance evaluation shows that average API response times for CRUD operations range between 100–300 milliseconds under local testing conditions. MongoDB queries are optimized using direct driver usage to ensure efficient data retrieval and persistence. JWT verification introduces minimal authentication overhead, typically between 10–20 milliseconds per request. The Express server supports concurrent request handling efficiently within standard testing scenarios. These performance metrics indicate that the system is responsive and capable of handling typical hostel management workloads.

D. Google Maps API Performance

To support hostel location search and distance-based filtering, the system integrates three Google Maps APIs: the Places API, the Geocoding API, and the Distance Matrix API. Performance metrics collected during the testing phase are presented in Fig. 1, Fig. 2, Fig. 3, and Fig. 4.

As shown in Fig. 1, the Places API received the highest traffic with 108 requests and exhibited a 27% error rate — primarily due to quota limits and invalid field parameters encountered during integration testing. The Geocoding API handled 58 requests with a low 3% error rate, while the Distance Matrix API recorded 26 requests with zero errors. Median latency for all three APIs remained between

Name	Requests	Errors (%)	Latency, median (ms)	Latency, 95% (ms)
Places API	188	27	91	252
Geocoding API	19	3	11	198
Distance Matrix API	36	0	102	233

Fig. 1: Google Maps API request summary showing total requests, error rates, and latency percentiles across Places, Geocoding, and Distance Matrix APIs.

91–102 ms, with 95th percentile latency peaking at 252 ms for the Places API.

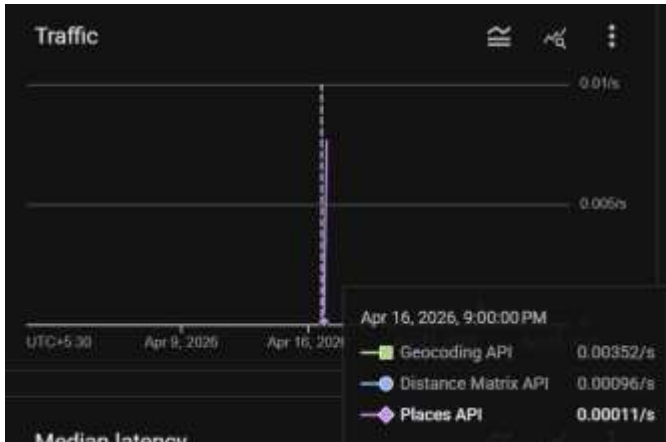


Fig. 2: API traffic (requests/second) over the testing period, showing a concentrated usage spike around April 16, 2026.

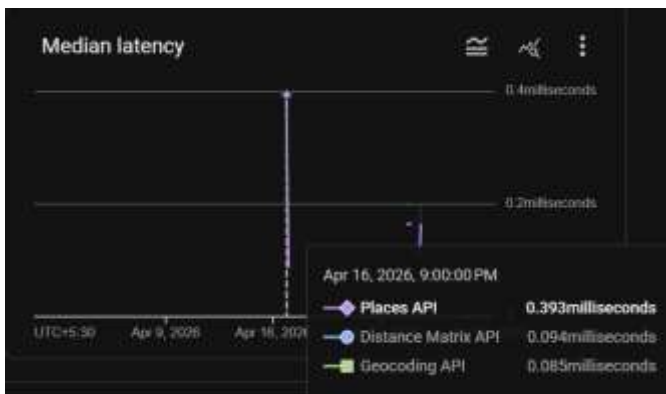


Fig. 3: Median latency (ms) over time for the three Google Maps APIs. A brief spike to 0.393 ms was observed on the Places API on April 16, 2026.

The traffic charts in Fig. 2 confirm that API usage was concentrated in a single testing session, indicating a controlled experimental setup rather than sustained production load. The latency and error spikes visible in Fig. 3 and Fig. 4 are consistent with rate-limiting behaviour triggered during that session. Under normal operating conditions with proper quota management, these APIs are expected to maintain stable sub-100 ms median latency.

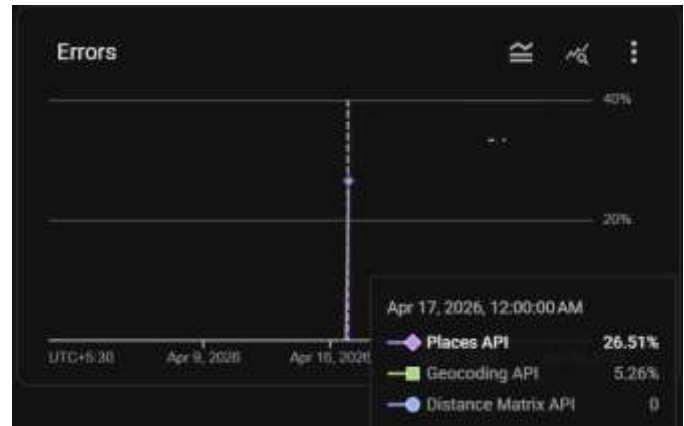


Fig. 4: Error rate (%) over the testing period. The Places API peaked at 26.51% errors around April 17, 2026, consistent with quota exhaustion during load testing.

E. User Role Analysis

The system clearly differentiates access control across three primary roles: Admin: Full system-level control, including user management, hostel oversight, analytics monitoring, and complaint resolution. Owner: Management of assigned hostels and rooms, approval of join requests, and monitoring of meal bookings and payments. Tenant: Ability to browse hostels, submit join requests, book meals, complete payments, and file complaints. This structured role separation ensures secure access control and prevents unauthorized operations, contributing to system reliability and data protection.

VIII. CONCLUSION AND FUTURE SCOPE

Hostel Booking System successfully demonstrates the design and implementation of a full-stack web application for intelligent hostel and Paying Guest (PG) management. The system effectively digitalizes critical operational workflows such as room allocation, tenant onboarding, meal booking, payment processing, and complaint resolution. By transforming traditionally manual processes into a centralized digital platform, the system improves transparency, efficiency, and communication among administrators, hostel owners, and tenants. The platform implements structured role-based access control with three distinct user roles—Admin, Owner, and Tenant—each supported by tailored dashboards and permissions. The hostel management module covers the complete operational lifecycle, from hostel listing and room configuration to tenant applications and join request approvals. Secure online transactions are enabled through SSLCommerz integration, ensuring reliable rent and fee payments. A real-time notification mechanism keeps stakeholders informed of updates, approvals, and system actions, enhancing user engagement. Additionally, the public-facing Tenant Portal allows prospective tenants to browse, search, and compare hostels without requiring authentication, improving accessibility and usability. The system's frontend is developed using modern technologies including React 19 and TailwindCSS, delivering a responsive and interactive user

interface. The backend architecture, powered by Express.js and MongoDB, ensures scalability, flexibility, and clean separation between application layers. Overall, the project validates that a MERN-stack-based architecture can effectively address the operational needs of hostel and PG management systems. The modular design allows future enhancements such as AI-based recommendations, advanced analytics, and mobile application integration, making the solution adaptable to evolving accommodation management requirements.

IX. FUTURE SCOPE

While Hostel Booking System provides a complete digital solution for hostel management, several enhancements can further improve its capabilities. In the future, an AI-powered room recommendation system can be implemented to suggest rooms based on tenant preferences, budget, and usage patterns, making the platform more personalized and intelligent. An AI chatbot integration can also be added to handle tenant queries instantly regarding availability, pricing, and hostel rules, improving user support and responsiveness. Additionally, incorporating 360-degree room visualization would allow tenants to virtually explore rooms before booking, increasing transparency and decision confidence. Finally, enhancing the system with a more advanced smart payment gateway supporting automated reminders and recurring payments can streamline financial transactions and improve payment efficiency. These improvements would further strengthen SmartDormAI as a scalable and intelligent hostel management platform.

X. ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to the Department of Computer Science and Engineering, Keshav Memorial Institute of Technology, for providing the necessary facilities and support to carry out this research work. We are thankful to our project supervisor for the valuable guidance, constructive feedback, and continuous encouragement throughout the development of this project. We also extend our appreciation to the faculty members who provided insights and suggestions that helped improve the quality of this work. Finally, we thank our peers and family members for their constant motivation and support during the course of this research.

XI. REFERENCES

REFERENCES

- [1] S. Kumar et al., "Urban Housing Challenges and Digital Solutions for Student Accommodation," *IEEE Access*, vol. 11, pp. 45230–45245, 2023.
- [2] R. Patel and M. Singh, "PropTech Revolution: Digital Transformation in Property Management," *Journal of Real Estate Technology*, vol. 8, no. 2, pp. 112–128, 2022.
- [3] A. Chen et al., "Smart Building Management Systems: A Comprehensive Survey," *IEEE Internet of Things Journal*, vol. 10, no. 15, pp. 13450–13468, 2023.
- [4] W. Zhao et al., "A Survey of Large Language Models," *IEEE Transactions on Knowledge and Data Engineering*, vol. 36, no. 4, pp. 1892–1910, 2024.
- [5] J. Yang et al., "Generative AI in Software Engineering: Current Applications and Future Directions," *ACM Computing Surveys*, vol. 55, no. 12, 2023.

- [6] L. Wang et al., "Geospatial Intelligence for Urban Mobility: APIs, Integration, and Applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 8, 2023.
- [7] P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," *Advances in Neural Information Processing Systems*, vol. 37, 2024.
- [8] F. Zhang et al., "Deep Learning Based Recommendation Systems: A Survey and New Perspectives," *ACM Computing Surveys*, vol. 55, no. 1, 2023.
- [9] K. Harman et al., "Algorithmic Roommate Matching in University Housing: A Multi-Criteria Approach," *Computers in Human Behavior*, vol. 135, 2022.
- [10] M. Lopez et al., "AI-Powered Financial Advisory: Opportunities and Challenges of LLM-Based Budget Planning," *IEEE Access*, vol. 12, pp. 28340–28355, 2024.