IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, May 2026
Impact Factor: 8.2

HostelDesk: A Web-Based Hostel Management System Using MERN Stack

Darpan[1] and Rajendra Singh[2]
[1] Department of Computer Science and Engineering
[2] Dean, Department of Computer Science and Engineering
Raffles University, Neemrana, Rajasthan, India
[1]darpansharma374@gmail.com, [2]rajendra.singh@rafflesuniversity.edu.in

Abstract: *University hostel management is a critical yet administratively demanding function involving attendance tracking, complaint resolution, and daily communication of information such as mess menus to hundreds of students. Traditional paper-based systems are inefficient, error-prone, and lack real-time visibility for both students and wardens. This paper presents "HostelDesk," a full-stack web-based Hostel Management System that digitizes and centralizes these operations through a role-based dual-portal architecture. The system integrates three core modules: a Complaint Management System allowing students to submit and track maintenance requests by category and priority, a Daily Attendance Tracking System enabling wardens to monitor student presence in real time, and a Digital Mess Menu Module providing students with always-accessible weekly meal schedules. The application is built on the MERN stack — MongoDB, Express.js, React.js, and Node.js — following a RESTful API architectural pattern secured by JSON Web Token (JWT) based authentication. A novel 3-layer role authorization mechanism ensures strict separation between the Student and Warden portals, preventing unauthorized access. Experimental evaluation across 20 manual test cases demonstrates 100% functional accuracy. The system reduces complaint resolution time, eliminates manual attendance registers, and provides students with digital access to mess information without physical visits to the warden's office.*

Keywords: Hostel Management System, MERN Stack, JWT Authentication, Complaint Management, Attendance Tracking, Mess Menu, React.js, Node.js, MongoDB, REST API

I. INTRODUCTION

University hostels house hundreds of students during their academic tenure and require efficient management of daily operations including student registration, attendance monitoring, maintenance complaint resolution, and timely communication of information such as mess menus and announcements. Traditionally, these operations have been managed through paper-based systems and direct personal interaction between students and wardens, which suffer from numerous inefficiencies in the modern digital age.

At Raffles University, Neemrana, students report facility issues — such as malfunctioning fans, AC units, water leakages, and internet connectivity problems — through informal channels including phone calls and personal visits to the warden's office. This creates a disorganized workflow for wardens who have no centralized record of complaints, their status, or their resolution history. Additionally, paper-based attendance registers make real-time headcounts impossible, and physical notice boards for mess menus are frequently missed by students.

This paper makes the following contributions:

- A complete full-stack web application for hostel management using the MERN stack
- A role-based dual-portal architecture with JWT-secured authentication for Students and Wardens
- An integrated Complaint Management Module with category, priority, and real-time status tracking
- A digital Attendance Tracking System providing wardens with live presence summaries

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, May 2026
Impact Factor: 8.2

- A digital Mess Menu Module with auto-seeding and inline warden editing
- Experimental evaluation demonstrating 100% accuracy across 20 functional test cases

II. LITERATURE REVIEW

A. Hostel Management Systems

Hostel management systems have evolved significantly over the past two decades. Early systems were desktop-based applications offering limited functionality such as student registration and room allocation, with poor scalability and poor user interfaces [1]. Patel et al. (2018) proposed a web-based hostel management system for Indian universities addressing room allocation and fee payment using PHP and MySQL, but the system lacked complaint management and mess menu features [2]. Kumar and Singh (2019) developed an ASP.NET-based system with a complaint module; however, it was limited to Windows environments, making cross-platform deployment complex [3]. Sharma et al. (2020) developed an Android application for hostel management allowing students to submit complaints via smartphones, but it was inaccessible from desktop browsers used by administrative staff [4].

B. Complaint Management Systems

Mishra and Gupta (2021) conducted a comparative study of complaint management systems in Indian educational institutions and found that digital systems resolved issues 65% faster than manual systems, with significantly higher student satisfaction due to increased transparency [5]. Key features identified as essential include complaint categorization, priority assignment, status visibility for complainants, and historical record maintenance — all of which are incorporated in HostelDesk.

C. MERN Stack in Web Development

The MERN stack has emerged as one of the most popular technology stacks for full-stack web application development. It offers JavaScript uniformity across all application tiers, MongoDB's flexible NoSQL document model, seamless JSON data flow between layers, and Node.js's event-driven non-blocking I/O model for high concurrency [6]. Verma et al. (2022) demonstrated that MERN stack applications achieve average response times 40% lower than traditional PHP/MySQL applications under concurrent institutional management workloads [7]. Aggarwal (2018) demonstrated that React.js-based applications outperform traditional server-rendered applications in user interaction responsiveness [8].

D. Comparison with Existing Systems

Table I compares HostelDesk with previously proposed hostel management systems across key functional dimensions.

Feature	Patel et al. 2018	Kumar & Singh 2019	Sharma et al. 2020	HostelDesk
Platform	Web (PHP)	Windows Only	Android App	Web (All Browsers)
Complaint Management	No	Yes	Yes	Yes
Attendance Tracking	No	No	No	Yes
Mess Menu Management	No	No	No	Yes
Role-Based Access	Partial	Partial	No	Yes
Real-Time Dashboard	No	No	No	Yes
Cross-Platform	Yes	No	No	Yes
JWT Authentication	No	No	No	Yes

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, May 2026
Impact Factor: 8.2

III. PROPOSED SYSTEM

A. System Architecture

HostelDesk follows a three-tier client-server architecture:

Presentation Layer: The React.js Single Page Application (SPA) running in the user's browser, handling all UI rendering, user interactions, form submissions, and client-side routing.

Application Layer: The Node.js and Express.js REST API server containing all business logic, authentication middleware, request validation, and data processing.

Data Layer: The MongoDB database accessed through the Mongoose ODM library, persisting all user accounts, complaints, attendance records, and mess menu data.

The Presentation Layer communicates with the Application Layer through HTTP/HTTPS REST API calls. The Application Layer communicates with the Data Layer through Mongoose, providing schema validation and query abstraction over the MongoDB driver.

B. Technology Stack

The complete technology stack used in HostelDesk is as follows: React.js 18 and React Router v6 for frontend UI and client-side navigation; Axios for HTTP requests; Node.js v24 and Express.js 4 for backend API server; jsonwebtoken for JWT generation and verification; bcryptjs for password hashing with salt rounds; MongoDB 8.3 as the NoSQL document database; and Mongoose 8 as the ODM providing schema validation and query abstraction.

Layer	Technology	Purpose
Frontend	React.js 18 + React Router v6	Component-based SPA with protected routing
Frontend	Axios + React Hot Toast	API communication and user notifications
Backend	Node.js v24 + Express.js 4	REST API server and middleware
Backend	jsonwebtoken + bcryptjs	JWT authentication and password hashing
Database	MongoDB 8.3 + Mongoose 8	NoSQL storage with schema validation
Users Collection	email (unique), role, password (hashed)	Stores all students and wardens
Complaints Collection	category, priority, status, wardenNote	Tracks all maintenance complaints
Attendance Collection	student (ref), date, status	One record per student per day
MessMenu Collection	day (unique), breakfast, lunch, snacks, dinner	Fixed 7 documents updated in place

C. Database Design

HostelDesk uses MongoDB organized into four collections. The **User Collection** stores all registered users with fields for name, email (unique), bcrypt-hashed password, role (student/warden), roll number, room number, phone, and creation timestamp. The **Complaint Collection** stores student complaints with fields for student reference, student name, roll number, room number, category (Fan/AC/Electricity/Plumbing/WiFi/Furniture/Pest Control/Other), title, description, priority (low/medium/high), status (pending/in-progress/resolved), and warden note. The **Attendance Collection** stores daily records with fields for student reference, date (YYYY-MM-DD), status (present/absent/on-leave), leave reason, and the marking warden's reference. The **MessMenu Collection** stores one document per day of the week with breakfast, lunch, snacks, and dinner arrays each containing items and serving times.

D. API Design

HostelDesk follows REST architectural principles. All API responses are in JSON format. The authentication endpoints include POST /api/auth/register (public), POST /api/auth/login (public), and GET /api/auth/me (authenticated).

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 9, May 2026 Impact Factor: 8.2

Complaint endpoints include GET /api/complaints (role-filtered), POST /api/complaints (student), PATCH /api/complaints/:id (warden), and DELETE /api/complaints/:id (warden). Attendance endpoints include GET /api/attendance (warden), GET /api/attendance/summary (warden), and POST /api/attendance (warden). Mess menu endpoints include GET /api/mess (authenticated), GET /api/mess/today (authenticated), and PUT /api/mess/:day (warden).

E. Authentication Mechanism

HostelDesk uses JWT-based stateless authentication. Upon login, the server validates credentials against the bcrypt-hashed password stored in MongoDB. On success, a signed JWT token is generated containing the user's MongoDB ObjectId, signed with a secret key stored in environment variables, with a 7-day expiration. The client stores the token in localStorage and attaches it as an Authorization: Bearer header to all subsequent requests. A server-side protect middleware verifies the token on every protected route. A wardenOnly middleware provides an additional authorization layer, rejecting authenticated non-warden users with a 403 Forbidden response.

F. Module Design

The **Complaint Management Module** allows students to submit structured complaints specifying category, title, description, and priority. Complaints are automatically linked to the student's profile, capturing name, roll number, and room number. Wardens access a centralized complaint list with filter options (All, Pending, In Progress, Resolved) and can update status, add resolution notes, or delete complaints. Status changes are immediately visible to the student.

The **Attendance Management Module** retrieves all registered students and displays their attendance status for the selected date. Wardens change any student's status (Present, Absent, On Leave) using an inline dropdown. Live summary metrics on the warden dashboard reflect real-time attendance counts.

The **Mess Menu Module** stores a complete weekly menu in the database. On first startup, the system auto-seeds a default seven-day menu covering breakfast, lunch, snacks, and dinner with serving times. Wardens edit any day's menu through an inline form. Students view today's highlighted menu card and browse the full weekly schedule.

IV. EXPERIMENTAL RESULTS

A. Testing Methodology

HostelDesk was tested using manual black-box testing. Twenty test cases were designed to cover all functional requirements, boundary conditions, and role-based access control scenarios. Each test case specifies input conditions, expected output, actual output, and pass/fail status.

B. Test Cases and Results

The 20 test cases covered: student and warden registration, valid and invalid login, duplicate email rejection, complaint submission and validation, warden complaint status updates (pending → in-progress → resolved), complaint deletion, attendance marking (present/absent/on-leave) with live summary count updates, student and warden mess menu access, warden mess menu editing, role guard enforcement (student redirected from warden URL), logout functionality, protected route access without login, and mess menu auto-seeding on a fresh database.

C. Results Summary

All 20 test cases passed successfully, yielding 100% functional accuracy. The system correctly handles all functional scenarios, boundary conditions, and access control requirements. No critical bugs were identified. The application performed consistently across Microsoft Edge and Google Chrome browsers. The mess menu auto-seeding mechanism eliminated the need for manual database initialization on fresh deployments.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, May 2026
Impact Factor: 8.2

D. Performance Observations

The Warden Dashboard makes four parallel API requests using Promise.all on initial load — fetching complaints, attendance records, summary statistics, and the mess menu simultaneously — minimizing total loading time. The 3-layer role authorization (JWT verification → role check → wardenOnly enforcement) adds negligible latency while providing robust access control.

V. CONCLUSION

This paper presented HostelDesk, a comprehensive full-stack web-based Hostel Management System built using the MERN stack for Raffles University, Neemrana. The system successfully addresses the core challenges of traditional hostel management by providing a centralized digital platform serving both students and wardens through role-specific portals secured by JWT-based authentication. The complaint management module enables students to submit and track maintenance requests without phone calls or physical visits. The attendance module provides wardens with real-time headcounts replacing error-prone paper registers. The mess menu module keeps students informed about daily and weekly meals through a digital, always-accessible interface. All 20 functional test cases passed, confirming the system meets its defined objectives. The application is production-ready with password hashing, JWT expiration, and role-based access control. Future work includes WebSocket-based real-time notifications using Socket.io, email and SMS alerts via Nodemailer and Twilio, photo attachments in complaints using Cloudinary, a leave application workflow, QR code-based attendance, a React Native mobile application, and multi-hostel support.

ACKNOWLEDGMENT

I would like to thank Dr. Rajendra Khatana sir, for guiding me throughout this project and always being available whenever I was stuck. Sir's feedback during development really helped me improve and think in the right direction. I am also grateful to my family and friends for their constant support throughout. This project would not have been possible without all of them.

REFERENCES

1. R. Patel, M. Shah, and A. Joshi, "Web-based hostel management system for Indian universities," International Journal of Computer Applications, vol. 182, no. 15, pp. 12–17, 2018.
2. S. Kumar and R. Singh, "ASP.NET based hostel management with complaint tracking," Journal of Information Technology and Management, vol. 11, no. 2, pp. 45–53, 2019.
3. V. Sharma, P. Gupta, and K. Mehta, "Android application for hostel management: A student-centric approach," International Journal of Mobile Applications, vol. 9, no. 4, pp. 88–96, 2020.
4. A. Mishra and N. Gupta, "Comparative study of complaint management systems in Indian educational institutions," Journal of Educational Administration, vol. 14, no. 3, pp. 201–215, 2021.
5. R. Verma, A. Singh, and B. Pandey, "Performance analysis of MERN stack vs PHP/MySQL for institutional management systems," International Journal of Web Engineering, vol. 8, no. 1, pp. 34–42, 2022.
6. S. Aggarwal, "Modern web development using ReactJS," International Journal of Recent Research Aspects, vol. 5, no. 1, pp. 133–137, 2018.
7. D. Flanagan, JavaScript: The Definitive Guide, 7th ed. O'Reilly Media, 2020.
8. K. Banker, P. Bakkum, S. Hawkins, D. Lear, and T. Mackey, MongoDB in Action, 2nd ed. Manning Publications, 2016.
9. E. Brown, Web Development with Node and Express, 2nd ed. O'Reilly Media, 2019.
10. Mongoose Documentation, "Getting started with Mongoose," 2024. [Online]. Available: https://mongoosejs.com/docs/
11. React Documentation, "React — A JavaScript library for building user interfaces," 2024. [Online]. Available: https://react.dev

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 9, May 2026
Impact Factor: 8.2

12. Express.js Documentation, "Express — Fast, unopinionated, minimalist web framework for Node.js," 2024. [Online]. Available: https://expressjs.com
13. MongoDB Documentation, "MongoDB Manual," 2024. [Online]. Available: https://docs.mongodb.com
14. jsonwebtoken npm package, 2023. [Online]. Available: https://www.npmjs.com/package/jsonwebtoken