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

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

Design and Implementation of a Smart Hostel Maintenance Request System

Adelowo, Opeyemi Joshua[1], Ene-Dede Precious Deinma[2], Ighoteme Favour Temitayo[2]
[1]Dept. of Information Technology, Babcock University, Ilishan-Remo, Ogun State, Nigeria
[2]Department of Computer Science (Information Systems), School of Computing, Babcock University, Ilishan-Remo, Ogun State, Nigeria

Abstract - University hostels play a critical role in student welfare, yet most institutions continue to rely on manual, paper-based systems for managing maintenance requests, resulting in delayed responses, poor communication, and declining living conditions that negatively affect students' academic performance and mental health. This project aimed to design and implement UniMaintain, a web-based Smart Hostel Maintenance Request System, with a prototype developed for Babcock University hostels, to address these inefficiencies through a structured digital platform. The system was developed using an Incremental Software Development Model employing a three-tier architecture comprising React.js for the frontend, Node.js with Express.js for backend processing, and PostgreSQL for relational data storage, with role-based access control supporting three user roles: students, administrators, and maintenance staff. System evaluation confirmed successful fulfilment of all functional and non-functional requirements, with API response times averaging under 300 milliseconds, responsive performance across devices, and usability testing indicating intuitive navigation with minimal training requirements. UniMaintain significantly reduces administrative workload, improves communication transparency between students and hostel management, and supports data-driven decision-making, demonstrating how affordable digital solutions can transform facility management in resource-constrained educational environments. Future work should integrate IoT sensor technology and machine learning for predictive maintenance, extend deployment to multiple hostels and institutions, and develop a mobile application version to improve accessibility for students who primarily use smartphones.

Keywords—*Facility Management; Hostel Maintenance; React.js; Smart Maintenance System; Web-based System*

I. INTRODUCTION

A hostel is an infrastructural facility providing affordable lodging for a specific group of people, often university students living away from home. Combined with a home-like setting, it provides physical, social, psychological, and philosophical environments that contribute to student wellbeing [1].

Maintaining such infrastructure is critical, as it provides students with security and caters to needs such as water supply, laundry services, and internet access. However, maintenance operations are often manually managed, leading to failures and numerous student complaints [2].

Many university hostels experience poor maintenance, erratic electricity supply, water shortages, and inadequate sanitation. In some instances, poor maintenance has caused serious incidents [3]. Students often feel administrators ignore their complaints, and these living conditions affect both academic performance and mental health [3].

The implementation of a Smart Hostel Maintenance System addresses these gaps by digitising hostel maintenance workflows, improving communication between administration and students, and enabling real-time status tracking of reported issues [4].

A. Problem Statement

Most facility documentation is done manually with no real-time tracking mechanisms. This leads to poor communication between management and hostel users, delayed feedback, and inefficiency in managing hostel properties [5]. These gaps increase administrative load and student discomfort, necessitating an effective digital solution [3].

B. Aims and Objectives

This project aims to build a Smart Hostel Maintenance Request System integrating facility documentation, room maintenance, and feedback features. The specific objectives are to:

- Design a web-based Smart Hostel Maintenance Request System;
- Develop and implement the system with role-based access for students, administrators, and maintenance staff;
- Evaluate the system against defined functional and non-functional requirements.

II. LITERATURE REVIEW

The origins of student accommodation can be traced as far back as the 5th century with Nalanda Mahavihara, widely regarded as the world's first residential university [8]. Despite this long history, many universities still carry out maintenance through paper-based forms, resulting in wasted time, errors, and untracked issues [7].

Studies have highlighted the need to replace outdated systems with automated maintenance platforms. Research on university dormitory structures found that maintenance efforts were constrained by funding and irregular scheduling, resulting in compliance issues and student dissatisfaction [10]. Early maintenance management was entirely manual and reactive; in Malaysian public universities, the majority of maintenance was corrective rather than preventive, with significant weaknesses in planning, personnel, and material availability [12].

Over time, experts documented a progression from manual book-keeping to systematic, preventative, and ultimately

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

predictive maintenance. Technologies such as Building Information Modelling (BIM), Computerized Maintenance Management Systems (CMMS), and Digital Twins now enable facility managers to plan and act on real-time data [13][9].

Web-based and cloud-hosted management systems have emerged to address inefficiencies in traditional hostel administration [24]. Studies document systems using PHP, MySQL, and HTML that allowed students to report issues, upload images, and track requests, with role-based access for students, administrators, and maintenance officers [24][25].

The reviewed literature reveals a clear gap: most existing systems focus on basic complaint handling without real-time automation or predictive capabilities. IoT-integrated and AI-driven systems offer promising results but remain cost-prohibitive for smaller institutions [19][20]. This gap motivates the design of UniMaintain as a cost-effective, web-based solution.

A. Summary of Related Work

S/N	Author(s) & Year	Title	Approach	Strengths	Gaps
1	Sanusi (2021)	Building Maintenance in Students' Hostels	Quantitative survey	Identified poor planning as key factor	Lacks digital integration
2	Au-Yong et al. (2023)	Prioritising Hostel Maintenance	Analytical framework	Condition-based prioritisation model	Relies on manual data
3	Dejaco et al. (2022)	Digital Transition in FM: BIM, CMMS	Case study	Introduces digital twin and data-driven decisions	Expensive; needs expertise
4	Bhardwaj et al. (2022)	Hybrid AI-IoT Hostel Management	System design & prototype	Combines automation and analytics	Requires costly IoT setup
5	Ahmad Fahmi et al. (2024)	Hostel Management System (PHP/MySQL)	System design & testing	Improved transparency and access	No automation; basic complaint only
6	Sangade et al. (2024)	Streamlining Hostel Management	PHP/MySQL development	Reduced paperwork	Security and scalability issues

Table I: Summary of Reviewed Literature

III. SYSTEM ANALYSIS AND DESIGN

This chapter details the system requirements, architectural decisions, and design specifications that guided the implementation of UniMaintain. System analysis focused on identifying requirements for students, administrators, and maintenance staff, while system design translates those needs into a technical specification.

A. Functional Requirements

The following functional requirements were identified:

- User Registration and Login: Students, administrators, and maintenance staff register and log in using unique credentials. The system validates user identity through JWT-based authentication and bcrypt password hashing.
- Maintenance Request Submission: Students submit requests by selecting an issue category (e.g., plumbing, electrical, furniture), providing a description, and optionally attaching an image.
- Request Tracking and Management: Students track request status (Pending, In Progress, Resolved). Administrators receive, sort, prioritise, assign, and update requests. Maintenance staff view and update tasks assigned to them.
- Notification System: Students receive real-time in-app notifications on status updates. Administrators receive alerts on new request submissions. Maintenance staff are notified when tasks are assigned.
- Admin Dashboard: Administrators manage user accounts, oversee all requests, assign tasks to maintenance staff, and access analytics on request categories, volume trends, and average resolution times.
- Security: Passwords are encrypted using bcrypt hashing; all API communication is secured through JWT authentication and role-based access control.

B. Non-Functional Requirements

- Usability: Simple, intuitive interface requiring minimal training for all user roles.
- Performance: System handles concurrent users with API response times under 300ms.
- Security: Encrypted personal data and secured API communication.
- Scalability: Platform supports growing user base without performance degradation.
- Reliability: Available 24/7 with minimal downtime for continuous issue reporting.
- Accessibility: Responsive web interface accessible on any browser, computer, or mobile device.

C. Proposed SDLC Model

An Incremental Software Development Model was adopted for UniMaintain. This approach breaks the system into specific development phases, each tested before the next is added. Core functionalities such as request submission and admin viewing were developed first, followed by iterative additions of notifications, media uploads, and analytics. This model accommodates evolving requirements and allows bugs to be caught early in each module.

D. System Architecture

UniMaintain adopts a three-tier client–server architecture separating the application into three logical layers:

- Presentation Tier (Frontend): Built using React.js with TypeScript and styled with Tailwind CSS and shadcn/ui. Responsible for rendering the user interface and managing user interactions across all devices.
- Application Tier (Backend): Built using Node.js and Express.js. Handles all business logic, API routing, JWT authentication, and request processing. Exposes a RESTful API consumed by the frontend.
- Data Tier (Database): PostgreSQL is used as the relational database management system, storing all user records, maintenance requests, notifications, assignments, and audit logs.

Communication between tiers is secured through JWT authentication and role-based access control, ensuring each user role (student, administrator, maintenance staff) accesses only features relevant to their function.

IV. IMPLEMENTATION AND RESULTS

This chapter presents the practical implementation of UniMaintain. The system was developed according to the three-tier architecture and integrates secure authentication, role-based

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

access control, real-time notifications via Socket.IO, and administrative analytics.

A. Development Environment

The system was developed using the following stack:

- Frontend: React.js with TypeScript, Tailwind CSS, shadcn/ui, and Vite as the build tool.
- Backend: Node.js with Express.js for RESTful API development; TypeORM for object-relational mapping.
- Database: PostgreSQL with indexing on frequently queried fields (user_id, status, created_at).
- Authentication: JSON Web Tokens (JWT) and bcryptjs for password hashing.
- Real-Time Communication: Socket.IO for WebSocket-based in-app notifications.
- Version Control / Testing: Git and GitHub; Postman for API endpoint testing.

B. Backend Implementation

The backend exposes RESTful API endpoints consumed by the React frontend. Major API routes include: POST /signup, POST /admin-signup, POST /maintenance-signup, POST /login, POST /requests, GET /requests, PATCH /requests/:id/status, GET /notifications, and POST /announcements. All endpoints include input validation and structured JSON responses. Error-handling middleware ensures graceful failure responses.

PostgreSQL tables include Users, Maintenance Requests, Notifications, Announcements, and Status Logs. Foreign key constraints enforce relational integrity between students and their requests. Role-based access control restricts students to their own requests, maintenance staff to assigned tasks, and grants administrators full system access.

C. Frontend Implementation

The landing page introduces the system and provides entry points for student registration, administrator registration, and login, highlighting the core features: Report Issue, Track Status, and Get Updates.

The student registration form captures full name, university email, matric number, level, gender, hostel name, room number, and password. Administrator accounts capture full name, email address, managed level, managed gender, and password. Maintenance staff accounts capture full name, email address, specialization, and password.

The student dashboard displays a personalised greeting, hostel and room information, submitted requests with a status progress tracker (Received → In Progress → Resolved), and a recent activity panel. Students submit new requests by selecting an issue category, confirming room number, and providing a detailed description.

The administrator dashboard displays key metrics including total requests, average resolution time, most common issue category, and completion rate. Administrators can filter requests by status, assign tasks to maintenance staff, mark requests as completed, and export data as CSV. An analytics module visualises request status distribution (donut chart), issue category distribution (bar chart), and weekly activity trends.

The maintenance staff dashboard displays pending and completed task counters, a tabbed task view (My Tasks and History), and a recent assignments panel. Staff update task progress from Pending through In Progress to Resolved.

The notification system provides real-time status update alerts, announcement alerts, timestamp tracking, and mark-as-read functionality. Socket.IO ensures instant delivery without page refreshes.

D. System Evaluation

Functional tests verified reliable registration, request handling, notification delivery, and role enforcement across all three user roles. API response times averaged under 300 milliseconds for standard operations. The system performed responsively across desktop, tablet, and mobile screen sizes. Informal usability feedback indicated intuitive navigation, clear status indications, and a low training threshold for new users.

V. SUMMARY AND CONCLUSION

A. Summary

UniMaintain was designed to solve the problems of ineffective hostel maintenance management in Nigerian universities, particularly at Babcock University. Traditional manual processes including paper logs, verbal complaints, and delayed responses lead to unresolved issues, student dissatisfaction, increased administrative burden, and deteriorating living conditions. An Incremental Software Development Model allowed gradual development and testing of each module before integration. The three-tier architecture—React.js frontend, Node.js/Express.js backend, and PostgreSQL database—successfully implements user registration with role-based access control for three user roles (students, administrators, and maintenance staff), maintenance request submission with image attachment, real-time request tracking, status update notifications, administrative announcements, and analytics dashboards.

B. Conclusion

UniMaintain successfully demonstrates how a well-structured web-based platform can replace manual hostel maintenance processes with a transparent, formal, and responsive digital system. Real-time tracking, automated notifications, centralised oversight, and analytical dashboards directly address the communication breakdowns and delayed repairs identified in the problem statement. The incremental development approach allowed potential problems to be identified early and resolved iteratively. The final prototype satisfies all functional and non-functional requirements including usability, security (JWT authentication and bcrypt hashing), and performance (responsive across devices). UniMaintain showcases how affordable digital solutions can transform facility management at educational institutions with limited resources, promoting academic focus and broader institutional quality in Nigerian universities.

C. Recommendations

Based on the findings and implementation experience, the following recommendations are made:

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

- Strengthen security with two-factor authentication (2FA), email verification on registration, and HTTPS enforcement in production.
- Optimize database queries and introduce a caching layer (e.g., Redis) to support more concurrent users during peak periods.
- Deploy on a cloud platform (e.g., AWS, Heroku, or DigitalOcean) for scalability beyond one institution.
- Implement multilingual support and WCAG-compliant accessibility guidelines for broader user demographics.
- Partner with Babcock University administration for pilot deployment and formal training workshops.
- Develop a mobile application using React Native to improve accessibility for students who primarily use smartphones.

D. Limitations

- The system was evaluated as a prototype in a controlled development environment without full-scale deployment across multiple hostels.
- The current maintenance staff module operates reactively; administrators manually assign tasks without automated prioritisation, intelligent routing, or IoT-based fault detection.
- Usability evaluation was based on functional testing and informal feedback rather than large-scale quantitative user studies.
- The online-only design requires stable internet access, which may be a constraint in Nigerian university contexts with unreliable connectivity.
- No long-term impact assessment was conducted on student satisfaction or academic performance across multiple academic sessions.

REFERENCES

[1] Kumar, “Exploring the Impact of Students Hostel Life on Academic Performance,” Journal of Ecohumanism, vol. 3, no. 8, 2024.

[2] Joshua, “Design and Implementation of a Student Hostel Management System,” Current Trends in Information Communication Technology Research (CTICTR), vol. 1, no. 47, 2024.

[3] Eferakhorho, “Assessing the State of University Hostel Accommodation: Challenges, Implications, and the Way Forward,” World Journal of Biology Pharmacy and Health Sciences, vol. 1, no. 106, 2025.

[4] Kumar, “StudentConnect: A Digital Solution for Managing Late Permissions and Maintenance Complaints in Hostels,” Premier Journal of Science, vol. 15, no. 1, 2025.

[5] Kaur, “Design and Implementation of an Integrated Hostel Management System with Role-Based Access for Students and Wardens,” Journal of Technology Management for Growing Economies, vol. 16, no. 1, 2025.

[6] Wikipedia, “Iterative and Incremental Development,” [Online]. Available: https://en.wikipedia.org/wiki/Iterative_and_incremental_development. [Accessed 18 Feb. 2026].

[7] K. Joel, “H-MAIN: Hostel Maintenance System,” Global Scientific Journal, vol. 4, no. 10, 2022.

[8] Wikipedia, “Student accommodation,” [Online]. Available: https://en.wikipedia.org/wiki/Student_accommodation. [Accessed 12 Mar. 2026].

[9] Emoh, “Assessment of Factors Affecting Effective Hostel Maintenance in Federal Polytechnic Oko, Anambra State, Nigeria,” International Journal of Civil Engineering, Construction and Estate Management, vol. 9, no. 1, 2021.

[10] K. Ayanlowo, “Development of an Automated Hostel Facility Management System,” Journal of Science and Engineering, vol. 5, no. 1, 2024.

[13] M. T. Hassanor, “Predictive Maintenance Integration with CMMS,” International Journal of Scientific and Research Publications, vol. 15, no. 11, 2025.

[18] A. M. A. Dompey, “Digital innovation in building maintenance,” International Journal of Building Pathology and Adaptation, vol. 3, no. 8, 2025.

[19] T. K. Singhal, “IOT Enabled Smart Hostel: A Futuristic Perspective,” IJRASET, vol. 5, no. 1, 2023.

[20] J. Mourya, “Leveraging the MERN Stack for IoT Applications,” International Journal of Applied Mathematics, vol. 38, no. 11, 2025.

[24] I. Eweoya et al., “Development of Web-Based Hostel Management System,” British Journal of Computer, Networking and Information Technology, vol. 8, no. 1, 2025.

[25] A. Kumar, P. Sharma and R. Verma, “Streamlining Hostel Management: An Innovative Online System for Efficient Administration,” International Journal of Computer Applications, vol. 185, no. 7, 2023.