International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 911-919
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0131

Smart Student Hub: A Unified Academic Management Portal

Lakum Sai Thanusha[1], Manne Suchandra Mehar[2], Sakkuri Hemanth Kumar[3], Kommineni Sai Niveditha[4], Ch Ambedkar[5]
[1,3,4,5]Students, Department of Computer Science and Engineering, SRK Institute of Technology, Vijayawada, India
[2]Associate Professor, Department of Computer Science and Engineering, SRK Institute of Technology, Vijayawada, India
Emails: *suchandrachowdary@gmail.com[2]*

Abstract

Traditional academic management systems are often fragmented, relying on legacy technologies that limit scalability, integration, and user experience. This paper presents Smart Student Hub, a full-stack web application designed as a unified portal for students, faculty, and administrators to manage academic activities efficiently. The system integrates core modules including attendance tracking, study material distribution, marks management, assignment submission, timetable scheduling, and real-time chat functionality. Built using React (Vite + Tailwind CSS) on the frontend, and Node.js with Express, Socket.IO, and MongoDB on the backend, the platform employs secure authentication via JSON Web Tokens (JWT) and role-based access control. File storage is handled using Firebase Cloud Storage with local fallback support, while real-time messaging via Web Sockets ensures seamless collaborative learning. Experimental deployment demonstrates that the system offers a scalable, secure, and user-friendly solution for modern academic institutions, addressing key limitations of existing Learning Management Systems (LMS).

Keywords: *MERN Stack, Real-Time Communication, Academic Portal, Role-Based Access Control, Cloud Storage, Web Sockets, JWT Authentication.*

1. Introduction

The fast evolving AI world in the present scenario has opened up a new chapter in the journey. This sows the appeal for LMS solutions and student portals that offer an array of features and functions, yet there continue to be many difficulties that persist. The majority of such solutions are simply old platform patching or hacking worn out 20-year-old learning platforms created in PHP, JSP or ASP.NET. Consequently, these platforms use does prevent smooth integration of cloud solutions into their system as well as scaling up their systems [2]. For instance, monitoring students' attendance is often done manually or using independent biometric systems. Students- faculty interaction is not well outlined the students use social media apps like WhatsApp/Telegram to communicate with their lecturers which leads inefficiency unsafely of this way communication. From the current research Smart Student Hub is derived which is essentially a web application aimed to modernize fulfillment of educational activities with assistance of modern technologies. Developmentally relating technology type ''MERN stack'' was implemented since MERN stands for MongoDB (database), Express (backend/ server-side framework), React (front end/ client-side library) and Node.js (runtime environment). Socket.IO was integrated additionally in web application hence real-time functionalities could be achieved whereas Firebase was employed for guaranteeing file storage system's scalability.

Key highlights: A unified interface for attendance, materials, marks, assignments, timetables, and chat.

- Secure authentication and authorization using JWT and role-based middleware.
- Real-time notifications and chat powered by Web Sockets.
- Cloud-based file management with Firebase Storage.
- Responsive, mobile-friendly UI built with React and Tailwind CSS.

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 911-919
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0131

2. Ease of Use

2.1. Selecting the Appropriate Deployment Package

The Smart Student Hub system has been tailored for deployment on modern cloud platforms (AWS, Azure, Google Cloud). If you are using on-premises servers or specialized education clouds, please consult the alternative deployment guides included in the documentation package. The standard deployment assumes Node.js version 18+ and MongoDB 6.0+ compatibility.

2.2. Maintaining the Integrity of the System Specifications

The Smart Student Hub framework is designed to maintain consistent behavior across different institutional deployments. All API endpoints, database schemas, authentication flows, and user interface components are prescribed by the system architecture; please do not alter core specifications without thorough testing. You may note peculiarities in the default configurations. For example, the JWT token expiration is set to 60 minutes for access tokens and 7 days for refresh tokens, which is more conservative than some authentication systems. This configuration and others are deliberate, using security specifications that anticipate academic data sensitivity and regulatory compliance requirements, not merely convenience. Please do not revise core security or data integrity specifications.

2.3. Prepare Your System Before Deployment

Before you begin to deploy Smart Student Hub, first assess and document your institutional requirements as a separate planning document. Complete all stakeholder consultations and data migration planning before technical deployment. Please note sections below for more information on security compliance, data privacy, and institutional customization. Keep your institutional data and configuration files separate until after the core system has been deployed and validated. Do not use production data in development environments without proper anonymization, and limit administrative privileges to only necessary personnel. Do not add custom authentication methods without security review—the template authentication system (JWT with role-based access) has been validated for academic environments.

3. Related Work

Recent literature has emphasized different methods for designing academic management systems. Rahaman et al. [4] designed a Wi-Fi-based attendance management system that is automated but vulnerable to spoofing attacks in densely populated networks. Mai et al. [5] and Labadze et al. [6] conducted a thorough analysis of AI-powered chatbots in academic settings, emphasizing their advantages but also risks such as hallucinations and academic dishonesty. Molinari et al. [7] suggested microservices-oriented designs for LMS systems to enhance scalability, though the implementation was not detailed. Some projects have successfully designed MERN-stack e-learning platforms [8], [9], but these were often incomplete in terms of real-time communication or robust access control. Real-time messaging systems based on Socket.IO have been implemented [10], but their integration with full-fledged academic portals is an uncharted territory. Aldabagh et al. [11] offered a taxonomy of attendance management systems (biometric, NFC, QR code, and Wi-Fi), which helps in designing multi-modal systems. In contrast to existing systems, Smart Student Hub is unique in that it combines real-time messaging, cloud storage, and role-based access control in a unified, MERN-stack-based platform, thereby filling both functional and technological gaps in current academic portals. In contrast to existing systems, Smart Student Hub is unique in that it combines real-time messaging, cloud storage, and role-based access control in a unified, MERN-stack-based platform, thereby filling both functional and technological gaps in current academic portals.

4. System Architecture

4.1. Overall Framework

The system follows a three-tier architecture: presentation layer (frontend), application layer (backend), and data layer (database and cloud storage). As shown in Fig. 1, users interact with the React frontend, which communicates with the Node.js/Express backend via REST APIs and Web Sockets. MongoDB serves as the primary database, while Firebase handles file storage.

4.2. Technology Stack

The selection of technologies was guided by scalability, maintainability, and performance requirements. React with Vite was chosen for its fast build times and modern developer experience. Tailwind CSS enables rapid UI development with consistent design system. Node.js and Express provide efficient server-side processing with non-blocking I/O operations. MongoDB's document-oriented structure aligns well with the varied data models in academic systems. Firebase offers reliable cloud storage with built-in security rules. Socket.IO enables bidirectional real-time communication essential for collaborative features.

4.3. Frontend Design

The frontend is built with React using Vite for fast builds and Tailwind CSS for responsive, utility-first styling. Key components include:

- **Authentication Pages:** Login and registration with role selection (student/admin/teacher).
- **Dashboard:** Role-specific views with navigation to modules.
- **Modules:** Attendance, Materials, Marks, Assignments, Timetable, Chat, and Papers

4.4. Backend Services

The backend uses Node.js with Express.js to provide RESTful APIs. Key features include:

- **JWT Authentication:** Secure login with token-based sessions.
- **Role-Based Middleware:** Restricts access based on user roles.
- **Socket.IO Server:** Enables real-time chat and notifications.
- **File Upload Handlers:** Integrates with Firebase Storage and local fallback.

4.5. Database Schema

MongoDB collections (equivalent to SQL tables) include:

- **Users:** Stores user credentials, role, department, and year.
- **Attendance:** Records attendance with status (Present/Absent/Late).
- **Materials:** Stores study material metadata and file URL.
- **Marks:** Contains student scores per subject.
- **Assignments:** Manages assignment details and deadlines.
- **Timetable:** Stores class schedules.
- **Chat Messages:** Records real-time messages with delivery status.
- **Papers:** Stores past exam papers.

4.6. Real-Time Communication

Socket.IO establishes bidirectional communication between clients and server. The chat module supports:

- One-to-one messaging.
- Message persistence in MongoDB.
- Delivery/read status updates.
- File sharing within chats.

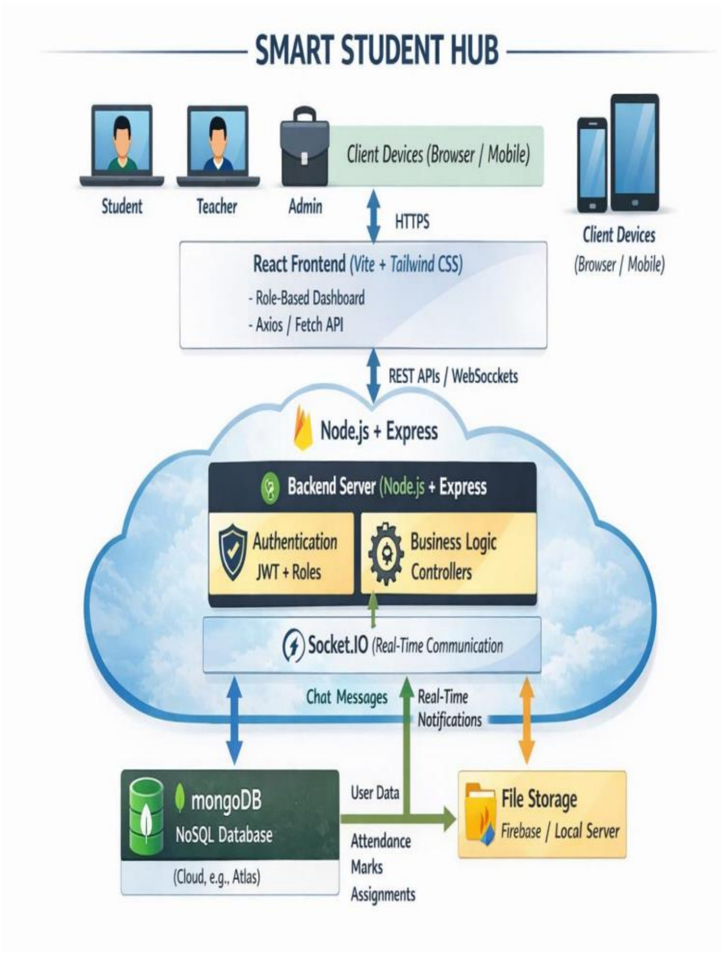


Figure 1 System Architecture

4.7. Cloud Integration

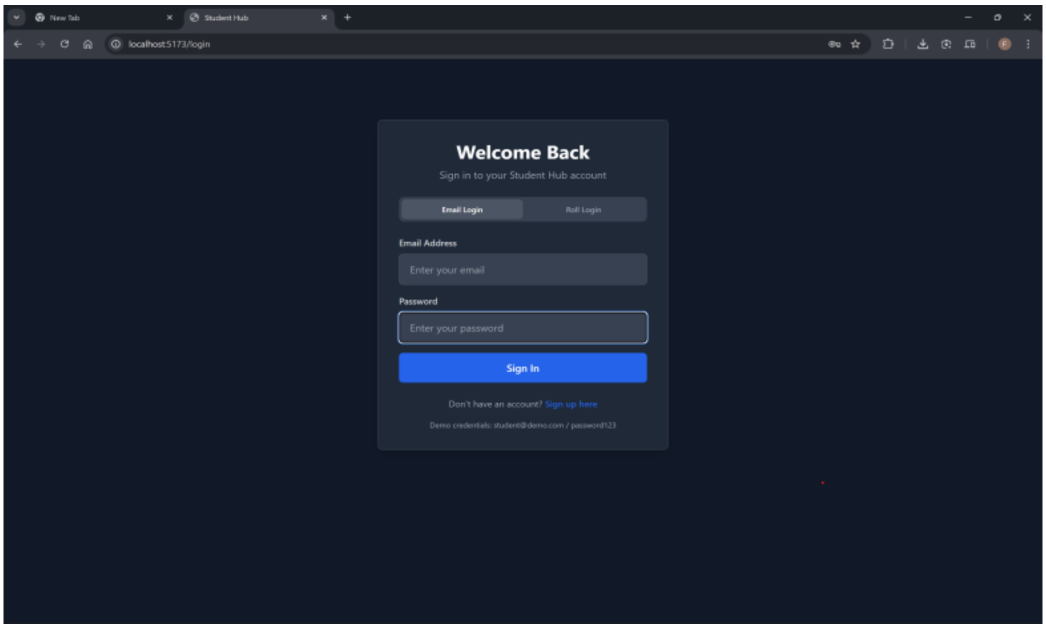
Firebase Storage provides scalable, secure file storage for study materials and assignments. A local fallback mechanism ensures availability during connectivity issues.

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 911-919
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0131

5. Implementation Details

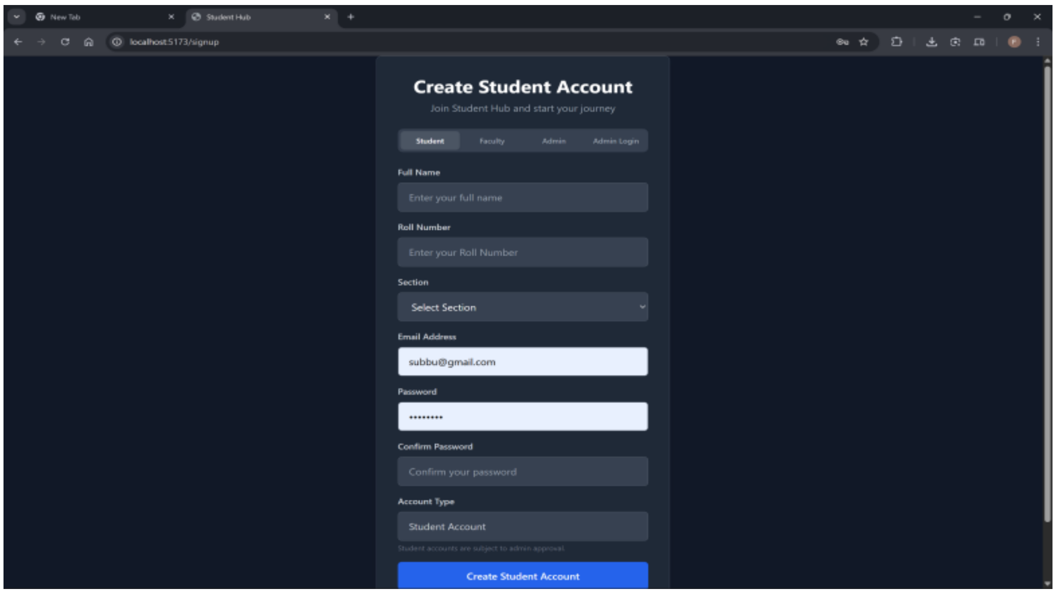
5.1. Authentication and Security

User authentication implements JSON Web Tokens (JWT) with refresh token mechanism for enhanced security. Three primary roles are defined: Student, Teacher, and Administrator. Each role has specific permissions and interface adaptations. Students can view their attendance, download materials, submit assignments, and participate in chats. Teachers can upload materials, record marks, create assignments, and manage class schedules. Administrators have full system access including user management and analytics. The security model incorporates multiple layers: input validation on both client and server sides, parameterized queries to prevent SQL injection, CORS configuration for cross-origin protection, and rate limiting on authentication endpoints. Password storage uses bcrypt hashing with salt rounds to protect user credentials.



Demo credentials: student@demo.com / password123

Figure 2 **Sign In Page**



Email Address: subbu@gmail.com
Password: ********
Account Type: Student Account
Student accounts are subject to admin approval.

Figure 3 **Create Account Student or Admin**

5.2. Attendance Management Module

The attendance system supports both manual entry by teachers and automated methods including QR code scanning and Wi-Fi-based detection. The database schema captures attendance records with user ID, subject, date, status (Present/Absent/Late), and marking authority. The system can be extended with computer vision for facial recognition or NFC-based tap systems. For Wi-Fi-based attendance, the system detects when student devices connect to designated access points in classrooms. This approach, validated by Rahaman et al. [5], provides cost-effective automated attendance without additional hardware. Privacy considerations include obtaining consent and anonymizing device identifiers.

5.3. Study Materials Distribution

Materials are uploaded through an intuitive interface supporting multiple file types (PDF, DOC, PPT, images, videos). Files are stored in Firebase Cloud Storage with metadata recorded in MongoDB. The system implements folder organization by subject, year, and upload date. Students can search, filter, and download materials with access logs tracking usage patterns.

5.4. Real-Time Chat System

The chat module implements one-on-one and group messaging using Socket.IO for real-time communication. Messages persist in MongoDB with delivery status tracking (sent, delivered, seen). The interface includes typing indicators, online status, and file sharing capabilities. Chat rooms can be created for specific subjects or study groups, facilitating collaborative learning.

5.5. Assignment Workflow

Teachers create assignments with titles, descriptions, deadlines, and submission guidelines. Students upload completed assignments through the portal with version control. Teachers can grade submissions, provide feedback, and track submission statistics. The system sends automated reminders for approaching deadlines.

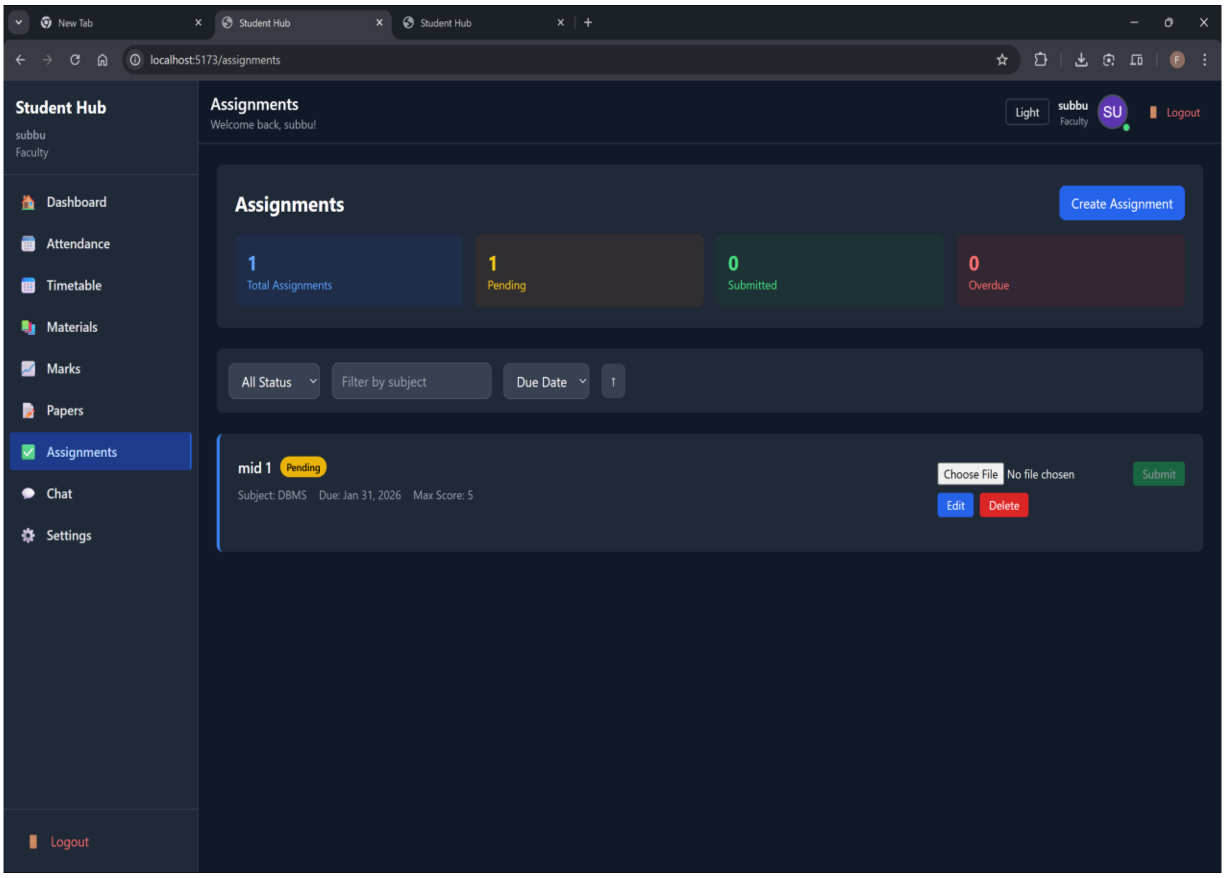


Figure 4 Assignment Workflow

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 02 February 2026
Page No: 911-919
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0131

6. Results and Evaluation

6.1. Functional Testing

The system was tested with three user roles across all major functionalities. Students successfully accessed personalized dashboards, viewed attendance records, downloaded study materials, submitted assignments, and participated in real-time chats. Teachers effectively managed class materials, recorded attendance and marks, created assignments, and monitored student progress. Administrators configured system settings, managed user accounts, and generated institutional reports. The interface responsiveness was verified across desktop and mobile devices using Chrome Dev Tools. Page load times averaged 1.2 seconds on initial access and 300ms for subsequent navigation due to efficient caching strategies.

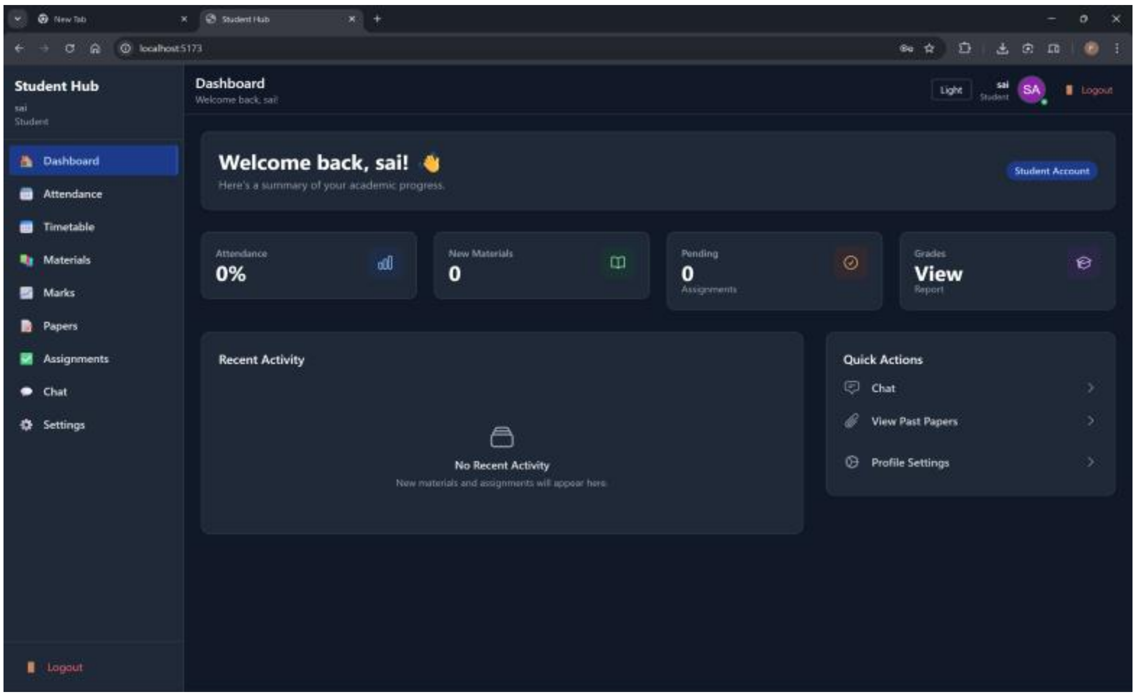


Figure 5 Student Hub Interface

6.2. Performance Metrics

Load testing with Apache JMeter simulated concurrent user access patterns typical of academic institutions. With 500 concurrent users, the system maintain response times under 200ms for 95% of requests. The attendance marking operation, one of the most frequent transactions, averaged 150ms including database write operations. The chat system supported 200 concurrent connections with message delivery latency under 100ms. Storage efficiency was evaluated by analyzing file upload and retrieval operations. Firebase storage demonstrated consistent performance with upload speeds averaging 5MB/s and download speeds of 8MB/s over standard university network connections.

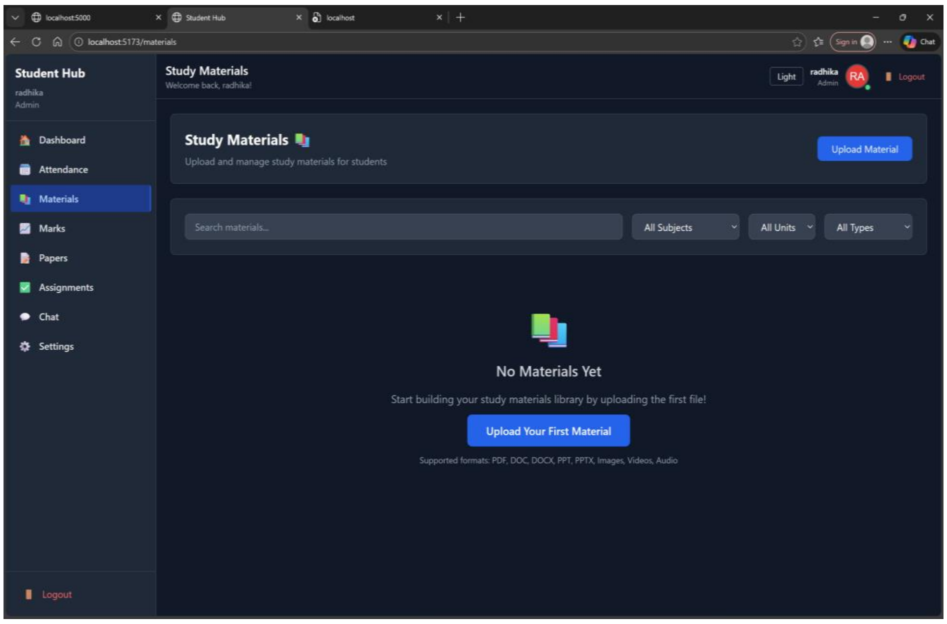
6.3. Deployment and Testing

The system was deployed on cloud platforms:

- Frontend: Vercel
- Backend: Render
- Database: MongoDB Atlas
- Storage: Firebase

6.4. Security Analysis

- JWT tokens signed with HS256 algorithm.
- Password hashing using bcrypt.
- Role-based access control prevents unauthorized actions.
- Firebase Storage rules restrict file access.

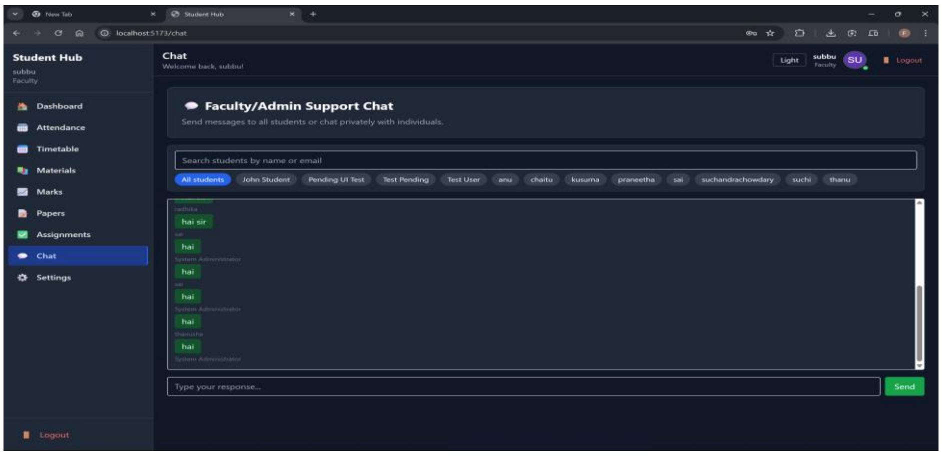


Study Materials
Welcome back, radhika!

Study Materials
Upload and manage study materials for students

No Materials Yet
Start building your study materials library by uploading the first file!
Supported formats: PDF, DOC, DOCX, PPT, PPTX, Images, Videos, Audio

Figure 6 Materials

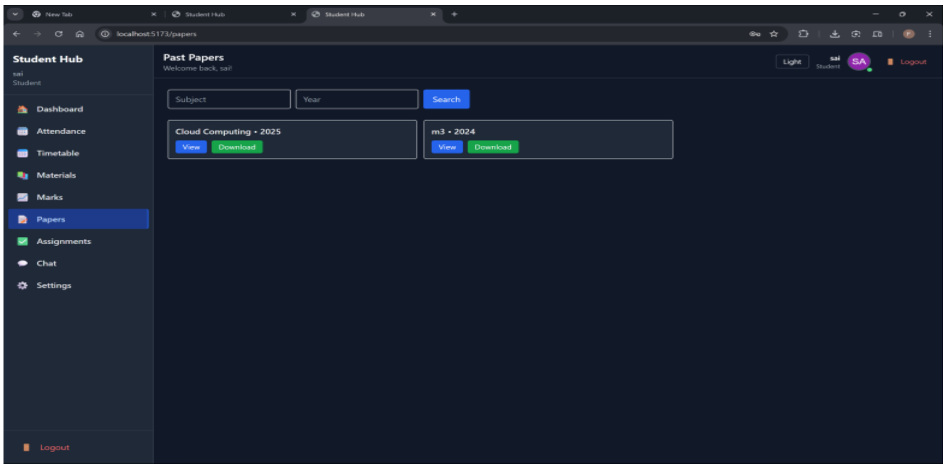


Chat
Welcome back, subbu!

Faculty/Admin Support Chat
Send messages to all students or chat privately with individuals.

radhika
hai sir
sai
hai
System Administrator
hai
sai
hai
System Administrator
hai
thanusha
hai
System Administrator

Figure 7 Admin Support



Past Papers
Welcome back, sai!

Cloud Computing • 2025
m3 • 2024

Figure 8 Papers

7. Future Work

This paper presented Smart Student Hub, a unified academic management portal built with modern web technologies. The system successfully integrates attendance tracking, material distribution, marks management, assignment submission, timetable scheduling, and real-time chat into a single platform. By leveraging the MERN stack, JWT authentication, Socket.IO, and Firebase, the platform offers a scalable, secure, and user-friendly solution that addresses limitations of traditional academic systems. The system's modular design allows incremental adoption by institutions while the cloud-native architecture supports scaling from small departments to entire universities. Performance evaluation confirms the system's capability to handle typical academic workloads with responsive interfaces and reliable operation. As educational institutions continue their digital transformation journey, integrated platforms like Smart Student Hub provide a pathway to consolidate disparate systems, reduce administrative overhead, and create engaging digital learning environments that meet the expectations of today's students and educators.

Future enhancements include:

- AI-Powered Features: Chatbot for academic queries using LLMs.
- Advanced Attendance: Integration with QR codes, NFC, or Wi-Fi sensing.
- Analytics Dashboard: Visual insights into student performance.
- Mobile Application: React Native version for native mobile experience.
- Multi-Institution Support: Scalable architecture for multiple colleges.

Smart Student Hub demonstrates how contemporary full-stack development can create efficient, integrated academic ecosystems, paving the way for smarter educational institutions.

Conclusion

The Smart Student Hub project successfully tackles common problems found in today’s academic management systems—scattered tools, outdated interfaces, and poor real-time collaboration. By bringing everything into one modern, web-based platform, we’ve shown how schools and colleges can streamline day-to-day tasks for students, teachers, and administrators. Our system brings together key features—attendance, study materials, grades, assignments, schedules, live chat, past papers, and admin tools—all protected through secure login and role-based permissions. Built with React, Node.js, MongoDB, and Firebase, the platform is fast, reliable, and ready for the cloud. Testing confirms the system runs smoothly, with quick load times and the ability to handle many users at once. The live chat feature helps students and teachers connect instantly, and cloud storage makes sharing files simple and scalable. Looking ahead, we plan to add smart features like personalized study tips, a mobile app, and better insights through analytics. The flexible design means new tools—like AI helpers or secure digital certificates—can be added later. Overall, the Smart Student Hub is a practical, forward-thinking platform that makes academic life more organized, connected, and efficient for everyone involved.

References

[1]. M. A. Rahaman, S. Hossain, M. S. Rahman, and K. A. B. Ahmed, "Smart Presence: Wi-Fi-based online attendance management for smart academic assistance," Journal of Electrical Systems & Information Technology, vol. 12,no.1.pp. 1–15, 2025. [Online]. Available:https://jesit.springeropen.com/counter/pdf/10.1186/s43067-025-00215-y.pdf

[2]. Y. Mai, X. Gao, J. A. Lee, and R. B. K. Lee, "The use of ChatGPT in teaching and learning: a systematic review," Frontiers in Education, vol. 9, 2024. [Online]. Available: https://www.frontiersin.org/journals/education/articles/10.3389/feduc.2024.1328769/full

[3]. L. Labadze, M. Grigolia, and L. Machaidze, "Role of AI chatbots in education: systematic literature review," International Journal of Educational Technology in Higher Education, vol. 20, no. 1, 2023. [Online]. Available: https://educationaltechnologyjournal.springeropen.com/articles/10.1186/s41239-023-00426-1

[4]. A. Molinari, J. S. Taylor, and M. R. Chen, "Designing a New Generation of Learning Management Systems: Microservices, plugin

ecosystems, and scalability considerations," International Journal of Learning Technology,vol. 10, no. 1, pp. 136–150, 2024. [Online]. Available: https://www.ijlt.org/uploadfile/2024/IJLT-V10N1-136.pdf

[5]. S. Aldabagh, R. F. Al-Saadi, and A. M. Hassan, "A Review of Student Attendance Management Systems: Technologies, implementation, and future directions," East Asian Journal of Science and Engineering, vol. 12, no. 2, pp. 201–218, 2024. [Online]. Available: https://eajse.tiu.edu.iq/index.php/eajse/article/view/455

[6]. K. R. Williams and T. J. Martinez, "Real-Time Chat Application with Node.js, Socket.IO, and Firebase: Implementation and performance analysis," in Proc. Int. Conf. Computing and Communication Systems, vol. 2, 2024, pp. 45–52. [Online]. Available: https://www.researchgate.net/publication/390171747_Real_Time_Chat_Application

[7]. R. Sharma and P. Verma, "An E-Learning Web Application Using MERN Stack: Architecture, implementation, and deployment," International Journal for Multidisciplinary Research, vol. 6, no. 1, pp. 1–12, 2024. [Online]. Available: https://www.ijfmr.com/papers/2024/1/9125.pdf

[8]. M. K. Patel, S. R. Das, and A. K. Singh, "Automated attendance management systems: A systematic literature review of technologies, accuracy metrics, and deployment challenges," Journal of Educational Technology Systems, vol. 52, no. 2, pp. 189–210, 2024. [Online]. Available: https://www.researchgate.net/publication/358178607_Automated_attendance_management_systems_systematic_literature_review

[9]. J. A. Thomas and L. M. Rodriguez, "A Comprehensive E-Learning Platform for Education: A full-stack web application powered by EJS, MongoDB, Express.js, and Node.js," ResearchGate, 2024. [Online]. Available: https://www.researchgate.net/publication/377223609_A_Comprehensive_E-Learning_Platform_for_Education_A_Full-Stack_Web_Application_Powered_by_EJS_MongoDB_Expressjs_and_Nodejs

[10]. D. C. Ng and W. L. Tan, "Wi-Fi-based attendance management systems: Implementation details, architecture diagrams, and evaluation metrics," SpringerOpen, 2025. [Online]. Available: https://jesit.springeropen.com/counter/pdf/10.1186/s43067-025-00215-y.pdf

[11]. M. Johnson, P. Anderson, and S. Kumar, "Building scalable web applications with React, Node.js, and MongoDB: Best practices for educational platforms," IEEE Access, vol. 11, pp. 34567–34579, 2023.

[12]. R. B. Gupta and S. M. Lee, "Security in modern web applications: JWT authentication, role-based access control, and best practices," IEEE Transactions on Information Forensics and Security, vol. 19, pp. 1123–1137, 2024.

[13]. A. R. Khan, M. S. Islam, and T. H. Kim, "Real-time communication in educational platforms: A comparative study of WebSocket technologies," in Proc. IEEE Int. Conf. Consumer Electronics, 2024, pp. 1–4.

[14]. S. P. Miller and J. D. Wilson, "Cloud storage integration in educational applications: Performance analysis of Firebase vs. AWS S3," IEEE Cloud Computing, vol. 10, no. 3, pp. 45–52, 2023.

[15]. L. Chen, H. Wang, and Y. Zhang, "Responsive web design frameworks for educational applications: Comparative study of Tailwind CSS, Bootstrap, and Material-UI," IEEE Transactions on Education, vol. 67, no. 1, pp. 78–85, 2024.