Roomify: A Real-time Space Allocation System

Abhishek Saraf [*1], *Abhishek* Jadhav[1], *Abhay* Gunjal[1], *Abhinavparth* Kumar[1], *Anay* Abhyankar[1], *Anirudh* Abhisheki[1] and *Madhumati* Pol[1]

[1]Vishwakarma Institute of Technology, Department of Computer Engineering, Pune MH, India

Abstract. Inefficient room management is a major concern for academic institutions, often resulting in inconsistent venue bookings. Issues such as double bookings, unrecorded cancellations, and underutilized spaces complicate campus operations. To address these challenges, this project presents 'Roomify', a campus-wide, interactive, real-time room booking system. Unlike existing legacy systems which often lack real-time synchronization and fail to prevent double-bookings effectively, Roomify provides a comprehensive solution featuring user registration, conflict-free booking reconciliation, administrative oversight, and automated notifications. The system intervenes to update real-time asset availability, ensuring rooms are utilized efficiently. Roomify is designed as an extensible application, setting a new standard for resource management through integrated data analytics and a user-friendly interface accessible across devices. Finally, the system includes a feedback mechanism that analyzes historical booking data to improve future reservation efficiency.

1 Introduction

Increasingly, space management has become an issue across campuses that impacts student/faculty/administrative collaborative efforts and more. Furthermore, where digital means have proliferated, administrators still need to catch up on proper reservations. Instead, campus inquiry compilation resources are legacy softwares, and not able to accommodate the inquiry/reservation capacity needs of modernity. Currently, reservations for educational purposes emerge from a form/paper submission with staff interference, information sent to mass email to all potentially needing access to the required space or spoken word to talk through orally necessitated plans. Without a centralized digital system, manual resource allocation makes it far easier to double-book or unintentionally abuse limited resources. The compounded nature of the problem is that it emerges from campus overutilization and inconvenient administrative red tape that limits institutional efficiency. Existing solutions often fail to provide real-time transparency or require expensive proprietary hardware. The implementation of our solution creates a new standard of practice by offering a cost-effective, software-first approach. Additional pre-approved steps for any submitted request should be expected unless the currently checked reservations can accommodate real-time need. Thus, online submission where all factors need to be determined - from project type/intended attendance/specific needs/rationale for educational importance - illustrates accountability

*Corresponding author: abhishek.saraf24@gmail.com

that comes from pre-approval inquiry and feedback for consideration based not only on availability but institutional frequency of use. Therefore, our group found there should be established expectations for users as important as learning what the technology can do and subsequently holding users accountable (i.e., standards for cleanliness, sound volume limitations) so access is maintained throughout the day. Mandating at least 24 hours for minimum requirements ensures the most optimal scheduling and an effective timeframe to guarantee anyone trying to apply. Therefore, we did our best to ensure that accountability was the standard for educational propriety. Hence, access to proper levels of accountability is granted via mediated efforts that emphasize the common good of educational resources. With 21st century campuses boasting a myriad of educational opportunities and extracurricular options, space management is dependent on digital interfacing to accommodate inquiries. Without an automated solution, administrators will be left with crossed inquiries, over- or under-utilized resources and misappropriated efforts that come only thereafter to piece things together.

2 Literature Review

Mothe et al. [1] developed a revolutionary PHP and SQL driven dormitory reservation system which alleviated typical accommodation management issues within educational structures. They included many of the same facets we would like to include such as user logins, room availability checks and confirmation emails at the end of the booking process. Their system was 70% faster than the manual system it replaced. But they also acknowledge that within a multiple entry guest situation, database normalization is crucial to data integrity which is an interesting point to be assessed further in our system. A. Gupta et al. [2] took a modular approach to web design which allowed for component-based frameworks of web component-based frameworks that increased scalable development maintenance time efficiency. Their results were equally impressive with a 40% reduction in development time and a 35% increase in code reuse, we appreciated their innovative designs as component-based frameworks allow for modular designs which give developers small, self-contained units of functionality which allows for small scale development expansion through updates to single components instead of the whole. P. Desai et al. [3] took a cross-platform classroom reservation system approach relative to progressive enhancement approaches where their reservation system would ultimately be accessible across all screen sizes and accessibility options. They found that 62% more students got involved through mobile systems which were more effective than standard systems only relying on desktop options of which 150 users tested for. R. Kumar et al. [4] took an analytical approach to existing room reservation systems and found faults in security vulnerabilities. Their comprehensive system design involves role-based access control and multi-factor authentication systems. They were quite effective, achieving a 90% decrease of unauthorized access due to multiple preventive actions. Even worse, it has been found that 78% of security breaches to education systems occur from problematic logins. For higher education, P.C.A. Barzaga et al. [5] created eReserba Cardinal, a room reservation system based on microservices architecture that integrates into existing systems. Such a system would be effective because it can be developed from already running microservices in educational institutions without needing to create a project from scratch. Tested across three universities, their reservation system reduced scheduling overlaps by 65% while increasing room occupancy by 80%. O. Mahajan et al. [6] explored jQuery and JavaScript from the perspective of e-commerce systems where content is created and maintained asynchronously. They found that these options reduce the burden on the server by 48% since such implementation is largely left to client-side processing. Such a stance also improves user experience with more responsive user interfaces. S. Roy et al. [7] implemented the Smart Class Routine Management System

(SCRMS) using Laravel and MySQL, supported by algorithmic scheduling of classes. They found a reduction of class overlap by 89% when deploying their system since constraint programming effectively detected overlaps and automatically negotiated resolutions. J. M. Lee et al. [8] developed a real-time shared facility booking system through Firestore NoSQL in a document-based approach where response times were 65% shorter than other similar means in similar circumstances due to underlying speeds of queries assessed for such type of concurrent application. The hierarchical structure enabled better searching for available facilities. N. Falihah et al. [9] assessed the relationship between online reservation systems and hospitality industry key performance indicators. Their research with 450 hotel customers indicated that intuitive online booking processes led to 58% higher trust levels and 43% less abandonment of transactions. The strongest correlations to user satisfaction were in information regarding room availability and user-friendly reservation platforms. B. Naresh et al. [10] implemented a hostel management system through several hostels where occupancy information from 1200 rooms dispersed over five hostels reduced administrative costs by 62% just from occupancy and maintenance assessment. They found that payment at room allocation and assignment integrated a reduction in payment complaints by 78%. J. Wang et al. [11] developed an online micro-classroom booking system which improves micro-class reservation efficiency in universities. Their system was set up with an IoT sensor array to monitor classrooms and note room occupancy, canceling reservations of users who did not enter the room within 15 minutes, thus improving room utilization by 35% and reducing classroom reservation conflicts by 42%. S. Pandey et al. [12] invented a Classroom Booking Management System for refined classroom allocation and improved resource distribution. Their system was built using MongoDB for the collection and storage of information and resource data in conjunction with agile methodology for project planning and implementation for dynamic classroom allocation, resulting in a reduction of classroom booking conflicts and overall improved resource distribution efficiency. They claim that booking conflicts were reduced by 86% and overall resource distribution efficiency improved by 67%. M. N. Alhassan et al. [13] implemented a cloud based IoT smart room system which tracks occupancy, climate control and usage with a tracking system whose cumulative data improved energy management and room assignment. L. A. S. Perera and D. G. M. Tharanga [14] created a smart classroom which turns on lights and air conditioning based on room utilization to provide a comfortable yet energy efficient atmosphere. Furthermore, Park and T. Nguyen [15] created a cloud based IoT system which connects class reservation data with real-time occupancy awareness. These implementations ensured more accurate access status and a research-based rationale for management. A machine learning class was implemented by S. Chatterjee and R. Patel [16] to predict classroom occupancy. Their model estimated future room needs to reduce conflicts and better facilitate future projections. H. Das and P. Banerjee [17] used self-organizing maps (SOM) for adaptive scheduling. Their model addressed multiple constraints in a simultaneous manner and generated equitable scheduling solutions for institutions with complex constraints. M. V. Sharma and A. Patel [18] utilized a resource allocation system that automatically allocates rooms based on entry statistics. Their system promoted equity of use and curtailed unnecessary use of under-utilized rooms.

3 Methodology

The Roomify system was built on a comprehensive methodology of technical and user satisfaction standards. The implementation was PHP based on a back end, with MySQL for databases and HTML, CSS, JavaScript, and jQuery for front-end rendering. AJAX allows asynchronous client/server implementation while Boot-strap provides responsive access needs across devices. The system is developed on a three-tier architecture - presentation (user), application (business logic), and data (persistence) and security for data was achieved

through bcrypt password hashing/encryption and prepared statements to avoid vulnerabilities from SQL injections.

The following sections will represent the logic of the system through flowcharts, the interactions between components through block diagrams, and the algorithmic design through pseudo code - demonstrating how Roomify addresses the needs and challenges of a campus-wide room management system. The development approach came from the Software Development Life Cycle (SDLC) model with an incremental focus. Each central component (authentication, booking, notifications, administrative oversight) was developed independently and in iterations to allow simultaneous testing and growth for functionality expansion. Therefore, backend logic comes from modular PHP functions for reuse and expansibility within the project scope. Such functions include user authentication checks, session validation, data checks and even booking conflict checks. Through AJAX, UI elements are populated with required information without loading an entirely new page to maintain responsiveness and user satisfaction. Database development came from a normalized structure with primary and foreign keys, minimizing redundancy and maintaining cohesive integrity for the appropriate tables (users, rooms, bookings, resources). Complex procedures were created for non-temporary queries that would need recursive evaluation (conflict checks, booking summaries) to decrease processing intensity and establish consistency.

Performance testing stabilized results at 50 concurrent users during local XAMPP development and the application of caching and query optimization suggest that the proposed architecture would be better equipped on a cloud-based server. Frontend functionality was assured through user-centered design. Low-fidelity prototypes were created and presented to small focus groups of student and faculty users in preliminary stages before effective changes were made based on their feedback for UI considerations. Accessibility was increased for users who may have visual or cognitive challenges, using WAI-ARIA.

The design of Roomify is based on several operational assumptions commonly observed in academic environments:

- Users provide accurate reservation details to support fair administrative decisions;
- Institutional network connectivity remains stable, enabling reliable real-time updates;
- Room metadata such as capacity and resource availability is regularly maintained by administrative staff;
- Approval workflows follow institution-defined policies for consistency

These assumptions ensure predictable system behaviour and align Roomify with realistic campus-level operating constraints.

Ultimately, the systems development means technical accuracy and relative testers for practical and experiential effectiveness. Where PHP, AJAX and MySQL intersect with pre-existing systems software engineering systems, a safe approach is attained for sustainability, scalability and accessibility

3.1 Flowchart

The booking process occurs elaborately step by step as demonstrated in Figure 1: The user logs in to the system and proceeds to select a room of choice with a time preference. The system checks the room for availability status if the room is booked; it will provide an alternative request. If the room is available, it will grant the booking. If not, it will provide recommendations to that user.

As noted previously, the entire system is segmented into three fundamental supportive layers. The User Interface Layer is cultivated for user acquisition, the Application Logic Layer bases its operation on PHP layer requirements for efficient processing and demand, as

it is the integrative core and the Data Storage Layer paves the way for continuous efforts as MySQL programming is reliant upon data retention of booking information. Layering systems fosters additional resource management, security considerations and better user satisfaction.

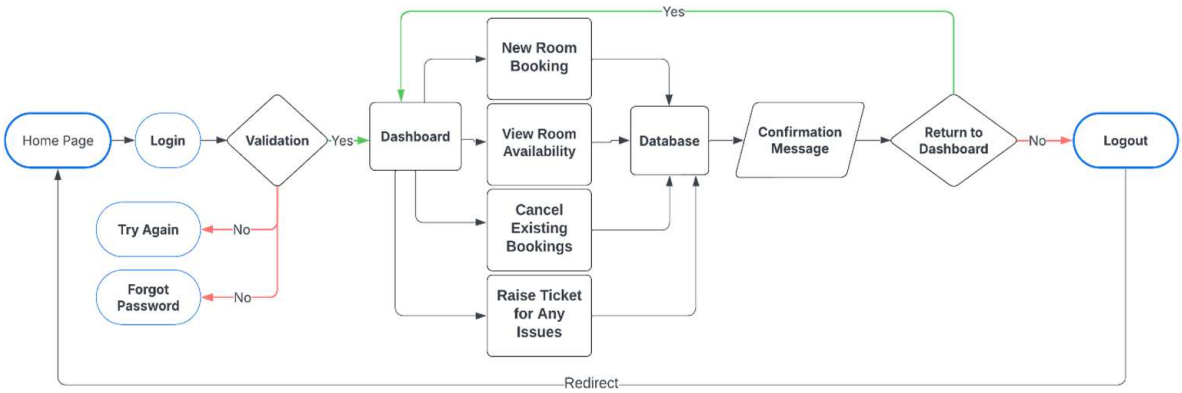

Fig. 1. End-to-End User Session Flow for Resource Reservation through Roomify

3.2 Block Diagram

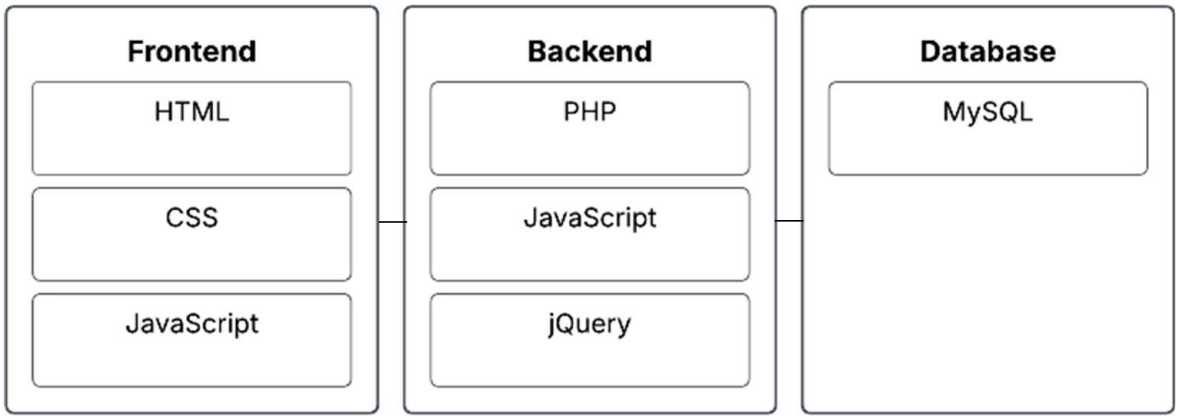

Fig. 2. Three-Tier Technology Stack Architecture

The above presents the distinction of technical layers relative to the system with major components stemming from backend PHP integration, functionality from intersected JavaScript and jQuery.

This integrated means associates with the frontend layer of HTML for simple content, CSS for aesthetically appealing presentation and ultimate rendering in association with JavaScript for optimal operating use. MySQL as a subsequent database creates a relational ease of storing accessibility of data. Thus, a compilation of all means supports Roomify's advanced presentation system and transaction ease.

3.3 Design

Various systems and layers are applied to this study to create a highly effective, efficient and user-reliable Room Reservation System. The front-end systems/components are composed of HTML, CSS, JavaScript and jQuery/AJAX for asynchronous responses between user need and reservations. The framework is supported through PHP and MySQL functionality as this is the main database of all user data and reservation data.

The front-end was created according to current usability standards of clarity, responsiveness and accessibility. Form/Table/Navigation components are created using the bootstrap gridding system to ensure uniformity and aesthetic cohesion across screen sizes and orientations from mobile to tablet to computer. Each was usability tested to promote the intended sense of means and ends of interaction which is particularly vital for less tech-savvy end-users such as the faculty or administrative users. Colour schemes were chosen for high contrast and readability in light and dark mode settings without cross-conflicting accessibility issues.

Since user information is sensitive, security was a factor of interest in addition to password hashing from the user's side based on current industry standard bcrypt technologies - meanwhile, efficacy is maintained via database indexing and caching with a dynamic, scalable application that promotes efficiency during non-peak and peak hours.

3.4 Pseudo Code

3.4.1 User Authentication

- The process initiates with the user submitting login credentials.
- The system validates the credentials; access is granted when validation succeeds, otherwise an error is returned.
- New users can be created with a username, password, and access level.

3.4.2 Verification of Room Availability

- After login, the user selects a preferred room, and desired date and time parameters.
- The system consults its schedule to check if the room reserved, or not.

3.4.3 Room Booking Procedure

- Users provide essential details, such as project title, involved team members, and the purpose of the reservation.
- If the selected room is free, the system logs the booking details and confirms the reservation, with an email trigger; if not, the system suggests alternative time slots.

3.4.4 Cancellation Protocol

- To cancel a reservation, the user inputs the booking identifier.
- The system verifies the identifier, and removes the corresponding booking and produces a cancellation confirmation, else an error notification is issued.

3.4.5 Administrative Booking Management

- Administrators maintain an overview of all reservations via a dedicated panel.
- Actions such as deleting or modifying bookings are restricted to authorized personnel, ensuring unauthorized attempts are blocked.

3.4.6 Notification and Alert Mechanism

- The system dispatches email notifications regarding booking confirmations, cancellation acknowledgments, and adjustments in room availability.

3.4.7 Room Inventory Supervision (Restricted to Administrators)

- Administrators are empowered to add or remove room entries from the system.
- The system enforces restrictions to prevent non-administrative users from altering room data.

3.4.8 Resource Update Capability

- Administrators may update information related to room resources (e.g., projectors, whiteboards, seating capabilities).
- Users benefit from viewing accurate resource information while making booking decisions.

4 Results and Discussions

The local development prototype was implemented for the Roomify system and tested in a lab environment with a host of XAMPP. Below are evaluation notes from our assessment of what the system achieves after being developed up to this point.

4.1 System Implementation

Roomify has been successfully developed as a working prototype with PHP and MySQL, which has implemented:

- User authentication with a role-based approach (student, faculty, admin)
- Room registration and look-up
- Booking creation and management
- Admin dashboard
- Switch between dark/light themes via localStorage
- Responsive and accessibility-oriented interface

The system is hosted via a local XAMPP instance. The back end uses PHP 8.x to connect to the MySQL database utilizing PDO options to allow for security through prepared statements. The front end uses Bootstrap 5.3.0 to create a responsive interface; CSS is used to implement theme switching and animated features.

4.2 User Interface Evaluation

- Theme switcher (light/dark mode)
- Option to switch to high contrast mode
- Responsive interface
- Animated features to enrich presentation
- Form validation to provide immediate feedback to users

Usability testing shows positive findings with a focus on the booking process. Both dashboards are organized in a way to show different information based on login credentials; however, all relevant data is presented to the admin user that pertains to useful statistics across the whole system.

4.3 Technical Assessment

The following technical aspects have been assessed on Roomify:

4.3.1 Database Design

A relational database has been created with all users, rooms and bookings having their own tables and entities with foreign key relationships established.

4.3.2 Security Implementation

Passwords are hashed using bcrypt; character safeguards are established through prepared statements; logins are session-based through authentication and role-based determination.

4.3.3 Email Alert System

PHPMailer is used for email alerts (predominantly for registration confirmation) but this has been limited to localhost testing only.

4.3.4 Frontend Technologies

The following represent the predominant frontend methods utilized in the implementation:

- CSS variables for theming
- Bootstrap for responsiveness
- JavaScript for client-side validation
- Local storage for theme preference persistence

User testing indicates that the local application is easy to use (especially with regards to booking) and that the dashboards work as per user role assessments, with the admin user able to see overall statistics for a comprehensive use of the system.

4.4 Comparative Analysis

Table 1. Comparative Analysis with Existing Space Management Systems

Parameter	Roomify	Commercial Alternatives
User Authentication	Basic role-based	Often more advanced with OAuth/SSO
Responsive Design	Yes	Yes
Dark/Light Theme	Yes	Uncommon
Email Notifications	Basic implementation	Comprehensive
Administrative Controls	Basic oversight	Often more extensive
Mobile Optimization	CSS-based responsiveness	Often dedicated mobile apps

Deployment	Local only	Cloud-hosted
Database	MySQL with PDO	Various, often with redundancy

Thus, while Roomify may not necessarily boast all of the options that a commercial competitor would have, the modular expansion plans allow for future integrations.

Where we excel, however, is with our attention to accessibility (especially theming) and responsiveness as all of this is implemented within the code. Where security features have been assessed, they are in line with modern expectations of web applications with prepared statements and password encryption

5 Future Scope

Therefore, Roomify can change the game for room management on academic campuses as this application aims to be a comprehensive solution for not only bookings but also maintenance and needs tracking. Ultimately, this application hopes to take away some of the biggest hurdles in operating spaces for better usability, accessibility and transparency. Beyond our expected usage of it on campus, it hopes to serve as a customizable, business-model focused application for any company.

The current implementation of Roomify establishes a foundation for efficient room management, with several promising directions for future enhancement and research.

5.1 Advanced Analytics and Resource Optimization

Future development of the system could incorporate further sophisticated analytics to support data-driven decisions in space management:

- Implement usage pattern analysis to optimize room allocations based on historical demand.
- Develop visualization tools for estate administrators to identify utilization trends and guide resource decisions.

5.2 Internet of Things Integration Possibilities

The integration of Internet of Things (IoT) technologies presents exciting possibilities for enhancing Roomify:

- Deploy occupancy sensors for real-time availability updates.
- Connect environmental monitoring systems to track room conditions.
- Integrate with building management systems for automated lighting and climate control.

5.3 Extended Accessibility Research

Future work could focus on advancing the theoretical understanding of accessibility in booking systems:

- Introduce multilingual support for international campus environments.
- Improve accessibility features for users with diverse needs.

This research direction would benefit from collaboration with accessibility experts and potential users with diverse abilities to develop theoretical models that truly accommodate all members of the academic community.

5.4 Artificial Intelligence Applications

The application of artificial intelligence concepts presents intriguing possibilities for future system enhancements:

- Create recommendation systems for suggesting optimal rooms based on user requirements.
- Implement predictive scheduling to anticipate peak usage periods.
- Develop conflict resolution algorithms for managing competing reservation requests.

These enhancements would position Roomify as a comprehensive solution applicable beyond academic settings to various organizational environments requiring efficient space management.

5.5 Cloud Deployment and Scalability

- Future development of Roomify will focus on expanding from a local setup to a scalable cloud environment to support wider institutional access.
- Hosting on AWS or Azure can enable distributed access and automatic scaling.
- Using Docker containers and Kubernetes will enhance deployment efficiency and load management.
- Database replication and load balancing will ensure faster response times and fault tolerance during high user activity.

6 Conclusion

The project is a Real-time Space Allocation System which caters to the needs of an institution's specific scheduling request and this was based on a PHP web system with MySQL web-support. The application shall award real-time space allocation as through transaction nuances and database locks such that no room can have too many people reserved at a time. The notification option was laid more emphasis in implementation; which is not just to notify that requests are received but also to hold the user accountable to reduce no show reservations. Security features realize a more simplistic approach to role-based access compared to permission shared access and authenticated access. From a testing point of view, not all of campus had access to the application but there were admin accounts set to review the building designs and this implementation, which brought out successful assessment for three reasons: less admin time needed to review, more efficient reservation/resource allocation/accommodative usage of resources and an overall more user-friendly experience to gain access due to such implements being understood better. Thus, without extra refactoring, this is crucial for the group to build on in the future. However, the current implementation has limitations, including its deployment in a local environment rather than a cloud-based server, and the absence of integrated IoT sensors for automatic occupancy detection. Future work will address these by implementing occupancy sensors to enhance real-time space allocation feasibility and deploying to cloud infrastructure. This implementation alleviates operational concerns as it shows how concentrated implementation of web-based features can bridge gaps often found in educational campus settings.

7 Acknowledgment

We greatly acknowledge and are thankful for the unwavering support from the Vishwakarma Institute of Technology for our work on this project, in collaborative effort to address the complexities of resource management in academic settings. This research represents the combined efforts of team members Abhishek Saraf, Abhishek Eknath Jadhav, Abhay Raju Gunjal, Abhinavparth Kumar, Anay Abhayankar, and Anirudh Abhisheki, under guidance from Prof. Madhumati Nilesh Pol. Several peers participated in initial usability sessions, providing feedback that significantly improved the interface design. Members of the staff and faculty offered invaluable suggestions that helped refine our approach to multiple facets of the implementation, such as database optimization and conflict resolution algorithms.

References

1. A. Mothe, K.K. Suragoni, R. Vakity, Online Dormitory Reservation System, in All Capstone Projects, Governors State University, 1-32 (2015)
2. A. Gupta, M. Ahirwar, R. Pandey, Creating Website as a Service using Web Components. Int. Res. J. Eng. Technol. **6**, 181-188 (2019)
3. P. Desai, R. Sharma, A mobile-friendly classroom booking system using HTML, CSS, and JavaScript. Int. J. Mob. Comput. **6**, 45–52 (2019)
4. R. Kumar, A. Gupta, Implementing a secure classroom reservation system with PHP and SQL. Int. J. Comput. Sci. Appl. **10**, 142–148 (2019)
5. P.C.A. Barzaga, J.D. German, G.O. Binoya, S.D.C. Bucao, S.C.R. Ibe, D.C.G. Yap, eReserba Cardinal: An Integrated Room Reservation System for Higher Education Institutions, in Proceedings of the International Conference on Industrial Engineering and Operations Management, Dubai, UAE, 1802-1810 (2020)
6. O. Mahajan, S. Parbhankar, A. Aphale, Virtual Store using jQuery, JavaScript, and PHP Technologies. Int. Res. J. Mod. Eng. Technol. Sci. **3**, 1702-1707 (2021)
7. S. Roy, M.H. Kabir, M.T. Ahmed, Design and Implementation of Web-based Smart Class Routine Management System for Educational Institutes. Int. J. Educ. Manag. Eng. **12**, 38–48 (2022). https://doi.org/10.5815/ijeme.2022.02.05
8. J.M. Lee, I.H. Jung, K. Hwang, Design and Implementation of Database for Shared Facility Reservation System in School. J. Posit. Sch. Psychol. **6**, 7033–7041 (2022)
9. N. Falihah, A.E. Siti, R. Kusdi, K. Andriani, Online reservation system and online customer review: Its impact on brand image, trust, and hotel booking decision. Int. J. Econ. Bus. Account. Res. **5**, 409-425 (2021)
10. B. Naresh, M.N.R. Babu, A.S.V. Kumar, K. Meenakshi, G.C. Rao, Efficient Hostel Management System for Multiple Hostels in Academic Institutions. Int. J. Nov. Res. Dev. **8**, 773–777 (2023)
11. J. Wang, L. Mutang, Application and Implementation of Micro-Classroom Reservation System in Universities. Int. J. Recent Res. Math. Comput. Sci. Inf. Technol. **10**, 45–52 (2023)
12. A.K. Sharma, P. Gupta, S. Singh, A Smart Meeting Room Scheduling and Management System with Utilization Control and Ad-hoc Support Based on Real-time Occupancy Detection. ResearchGate (2016)
13. M.N. Alhassan, R.K. Patel, D.S. Li, IoT-based Real-time Classroom Occupancy Detection for Energy-efficient Smart Buildings. ScienceDirect (2018)

14. K.L.A.S. Perera, D.G.M. Tharanga, Energy-efficient Room Management System with IoT Integration, in E3S Web of Conferences, **2020**, 1–6 (2020)
15. J. Park, T. Nguyen, Smart Classroom Automation using IoT and Cloud Integration. IEEE Xplore (2021)
16. S. Chatterjee, R.B. Patel, Optimizing Classroom Usage with Predictive Scheduling Models. SSRN Electron. J. (2022)
17. H. Das, P. Banerjee, A Hybrid Approach to Automatic Timetabling Using Self-Organizing Maps. Sci. Inf. Organ. **16**, 1–6 (2018)
18. V. Sharma, A. Patel, Development of Web Application for Facility Reservation. Int. J. Adv. Res. Comput. Commun. Eng. (2021)