IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Department Resource Management System

T. VINOTHINI #1,N. SAMBATHKUMAR #2,P. VIKRAM #3,S.B. SRIHARI #4,

#1 Professor, Adhiyamaan College of Engineering(An Autonomous Institution), Hosur

#2,3,4 UG Students, Adhiyamaan College of Engineering(An Autonomous Institution), Hosur

Abstract: The Department Resource Booking System is a comprehensive web-based platform developed to automate and streamline the management and allocation of institutional resources across multiple departments, eliminating manual booking inefficiencies through a centralized interface for tracking resource availability and usage. Built using the Django framework as the backend and an SQL database for secure and scalable data management, the system ensures real time synchronization, data consistency, and accessibility for authorized users. It adopts a role-based access model with three user categories — Staff, Lab In-charge, and Admin — each tailored to specific operational needs. Staff members can browse available rooms and categorized resources, view real-time availability, and submit booking requests for required quantities; Lab In charge users serve as intermediaries who validate resource availability, review staff requests, and approve or reject them before forwarding to the Admin; while the Admin holds full control, reviewing all forwarded requests and issuing final approvals or denials. To promote efficiency and transparency, the system includes automated notifications that keep users informed at every stage of the approval process. Furthermore, all user information and resource details can be dynamically managed within a centralized database, ensuring flexibility, scalability, and ease of maintenance. By integrating Django's robust framework with a structured SQL schema and secure authentication mechanisms, the Department Resource Booking System provides an efficient, transparent, and scalable digital solution for effective institutional resource allocation and management.

1. INTRODUCTION

1.1 Overview the Department Resource Booking System is an integrated web-based application developed to simplify and digitalize the management of institutional resources across various departments. The system addresses the challenges of manual record-keeping, inefficient communication, and lack of transparency in resource allocation by providing a centralized and automated platform. It allows users to efficiently request, approve, and monitor the usage of resources such as rooms, equipment, and materials within the institution.

Built using the Django framework and SQL database, the system ensures secure data handling, real-time updates, and easy scalability. The role-based access model divides functionalities among three user types — Staff, Lab In-charge, and Admin — ensuring clear workflow hierarchy and controlled access. Staff members can view available resources and submit requests, Lab In-charge users validate and process those requests, and the admin finalizes approvals or rejections.

Through this structured workflow, the system promotes accountability and transparency in resource management. Additionally, the inclusion of automated notifications, centralized data storage, and editable records ensures that all departments remain informed and up to date. Overall, the Department Resource Booking System enhances operational efficiency, reduces administrative workload, and supports data-driven decision-making within institutional environments.

1.2 Objectives The specific objectives are: To automate the resource management process by replacing manual booking and approval methods with a centralized, digital platform for improved efficiency. To implement a role-based access system that provides secure and distinct functionalities for Staff, Lab In-charge, and Admin users. To ensure real-time tracking of resources by maintaining updated availability records and enabling transparent request handling across departments.

To facilitate an efficient approval workflow where requests move systematically from Staff to Lab In-charge and then to the admin for final decision-making. To provide a centralized database system for storing and managing user, resource, and transaction details securely and consistently. To enhance communication and accountability through automated notifications and activity logs, ensuring clarity and transparency in every stage of resource allocation.

2. LITERATURE SURVEY

Efficient resource management systems have become essential in academic and institutional environments to optimize the use of shared facilities and assets. Several studies and existing systems have explored automation, database integration, and workflow transparency to enhance departmental operations.

Automated Resource Management Systems: Previous research highlights the limitations of traditional paper-based resource booking methods, which are prone to delays and errors. Automation using web-based tools significantly improves accuracy and response time.

Role-Based Access Control (RBAC): According to studies in information systems security, implementing RBAC ensures data confidentiality and integrity. By assigning different access levels to Staff, Lab In-charge, and Admin, the system prevents unauthorized modifications.

Database-Centric Resource Tracking: Research on database-driven systems emphasizes the importance of structured data storage for scalability and consistency. SQL-based databases provide a robust backbone for managing institutional resources efficiently.

Workflow Automation and Request Validation: Studies in process automation show that multi-level approval workflows enhance organizational transparency. A structured request validation chain reduces redundancy and prevents double allocation of resources.

Web-Based Platforms for Institutional Management: Literature suggests that web applications provide better accessibility and usability compared to standalone systems. Django-based implementations offer scalability, modularity, and a secure authentication mechanism.

Notification and Feedback Mechanisms: Prior systems emphasize the role of automated notifications in maintaining workflow transparency. Instant alerts ensure all stakeholders are informed about approval status and resource availability.

3. SYSTEM ANALYSIS

3.1 Existing System The current resource booking process in most departments is manual and paper-based, requiring physical forms and in-person approvals. Communication gaps often occur between staff, lab in-charge, and administration, leading to delays in processing requests. Tracking of resources and availability is done through spreadsheets or handwritten logs, making updates prone to human error. Lack of centralized data storage causes inconsistencies, as each department may maintain separate records of the same resources. No real-time visibility into resource status makes it difficult for staff to know whether items are available or already booked. Approval and feedback mechanisms are slow and inefficient, relying heavily on physical meetings or email exchanges. Limited accountability and transparency exist in the current system, as it lacks automated tracking and proper record management.

Disadvantages: The manual process is time-consuming and prone to human errors. Lack of real-time updates leads to double bookings and confusion. Inefficient communication causes delays in approval and allocation. Data stored in separate records results in inconsistency and data loss. Limited transparency and accountability make tracking resource usage difficult.

3.2 Proposed System The proposed system introduces a web-based platform to automate and streamline the entire resource booking and approval process. It provides role-based login access for Staff, Lab In-charge, and Admin, ensuring secure and structured workflow management. Staff members can view available resources, check their quantities in real time, and submit requests through a user-friendly interface. The Lab In-charge can review, verify, and approve or reject requests before forwarding them to the admin for final decisions. The admin has full control over managing users, resources, and final approvals, ensuring transparency and accountability. The system includes automated notifications and status updates, keeping all users informed about request progress at every stage. A centralized SQL database ensures consistent data storage, easy retrieval, and prevention of duplication or data loss. Built on the Django framework, the system ensures scalability, security, and efficient performance for institutional resource management.

Advantages: The system automates the entire resource management process, reducing manual effort and saving valuable time for staff and administrators. Real-time tracking of resources ensures accurate availability status and prevents double booking or over-allocation. Role-based access control enhances data security by restricting system operations according to user privileges. Centralized data storage maintains consistency across departments and simplifies data retrieval for reporting and audits. Automated notifications and updates improve communication and transparency between Staff, Lab In-charge, and Admin users. User-friendly web interface makes the system easy to use, accessible from anywhere, and reduces dependency on physical paperwork. Efficient approval workflow streamlines request validation and decision-making, ensuring faster and more reliable resource allocation.

3.3 Proposed Solution The proposed system aims to digitize and automate departmental resource management through a centralized, web-based platform. It introduces secure, role-based access for Staff, Lab In-charge, and Admin to ensure controlled and efficient system operations. Staff users can view available resources in real time, select required quantities, and submit booking requests easily. The Lab In-charge can review and verify requests, ensuring that only valid and available resources are forwarded to the Admin. The admin has full authority to approve or reject final requests, manage users, and oversee the entire booking process. The system integrates automated notifications and status tracking to keep all users informed about request updates instantly. Developed using Django and SQL database, the system ensures scalability, data integrity, and reliable performance across departments.

3.4 Problem-Solution Fit Problem: Manual resource booking causes delays and errors in allocation. Solution: A web-based automated system streamlines requests and approvals, reducing processing time and mistakes.

Problem: Lack of real-time visibility leads to double bookings and resource conflicts. Solution: The system provides live updates on resource availability, ensuring accurate and conflict-free allocations.

Problem: Communication gaps between Staff, Lab In-charge, and Admin slow down request processing. Solution: Automated notifications and a structured workflow improve coordination and timely decision-making.

Problem: Dispersed data storage creates inconsistencies and difficulty in record-keeping. Solution: A centralized SQL database ensures consistent, secure, and easily retrievable records for all users.

Problem: Lack of accountability makes it difficult to track resource usage and approvals. Solution: The system logs all activities and approvals, enhancing transparency and traceability of actions.

3.5 Architecture Design The system follows a client-server model where the frontend interacts with users via web browsers. Requests from Staff, Lab In-charge, and Admin are sent to the server for processing. This design ensures centralized control and efficient handling of multiple user requests simultaneously. The architecture incorporates role-based access to segregate functionalities for different users. Staff, Lab In-

charge, and Admin have specific permissions, ensuring security and controlled operations. A centralized SQL database stores all resource details, user information, and request logs. It enables consistent data retrieval, real-time updates, and seamless reporting. The database layer ensures data integrity, reduces redundancy, and supports scalability.

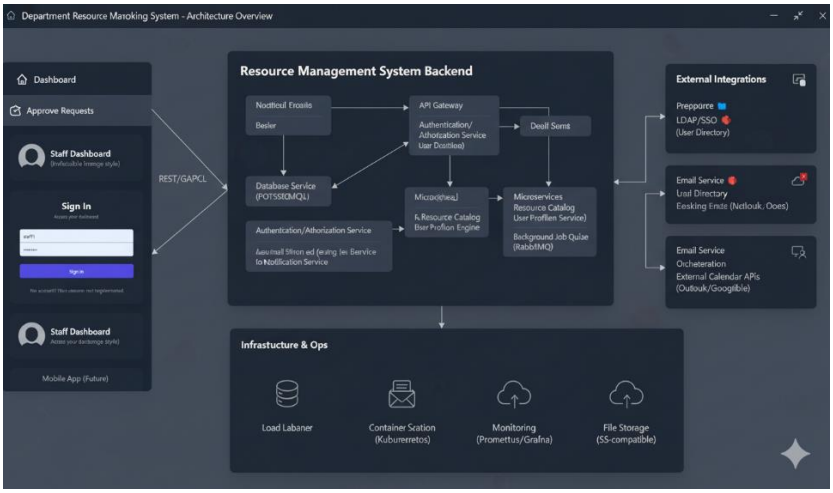

figure 1: System Architecture of Department Resource Management System

Data Flow Example (Application Submission): When a staff member needs a resource, they first log in to the system using their credentials. They browse available resources, select the required items, and submit a detailed request with quantity and purpose. This request is then stored in the centralized SQL database for further processing. Once submitted, the Lab In-charge receives a notification about the new request. They review the requested resources, check availability, and either approve or reject the request based on current stock. If the Lab In-charge approves the request, it is forwarded to the admin for final authorization. The admin evaluates the request, ensuring compliance with departmental policies and overall resource management strategies.

After the Admin makes a decision, the staff member receives a notification of approval or rejection. Approved requests are scheduled for allocation, and the database is updated to reflect the resource status.

3.6 DESCRIPTION OF MODULES

3.6.1 Staff Module the Staff module is designed for regular users who require access to departmental resources. Staff members can log in securely, view available resources, and check real-time availability. They can submit resource requests specifying the type and quantity of items needed. Each request is automatically timestamped and stored in the centralized database. Staff can also track the status of their requests and receive notifications regarding approvals or rejections. This module ensures that staff can efficiently plan and manage their resource needs.

3.6.2 Lab In-charge Module the Lab In-charge module serves as an intermediary between Staff and Admin. Lab In-charge users can view all submitted requests and validate resource availability before forwarding them for final approval. They have the ability to approve, reject, or request modifications to a request based on stock levels or departmental policies. The system provides automated notifications to staff regarding any updates or decisions. The module maintains detailed logs of all actions to ensure accountability. This module streamlines resource verification and prevents overbooking or misuse.

3.6.3 Admin Module the Admin module provides full control over the entire resource management system. Admins can manage user accounts, update resource details, and oversee all forwarded requests. They have the authority to approve or reject requests and generate reports on resource utilization. Notifications are automatically sent to Staff and Lab In-charge users to inform them of the status of requests. This module ensures centralized governance, policy compliance, and efficient allocation of institutional resources.

3.6.4 Resource Management Module the Resource Management module handles the cataloguing and tracking of all institutional resources. It maintains information on resource type, quantity, availability, and usage history. The system updates availability in real time as requests are approved or resources are allocated. Admins can add, remove, or modify resource details to reflect current inventory. This module supports accurate planning and prevents resource shortages. Overall, it ensures effective monitoring and optimal utilization of all assets.

3.6.5 Notification and Workflow Module the Notification and Workflow module automate communication between users at every stage of the request process. When a request is submitted, approved, or rejected, the system generates automatic notifications for relevant users. The workflow engine ensures that requests pass sequentially from Staff to Lab In-charge and then to Admin. This module maintains logs of all actions for accountability and transparency. It enhances user engagement by keeping everyone informed in real time. By streamlining the process, it reduces delays and improves operational efficiency.

3.7 DATA FLOW DIAGRAM

The Data Flow Diagram (DFD) visually represents the flow of information within the Department Resource Management System, showing how requests are submitted, verified, and approved. The system operates at two conceptual levels: a high-level context and a detailed process flow.

At the high-level context (Level 0), the system interacts with three primary external entities. Staff members submit resource requests and receive notifications about their status. The Lab In-charge reviews requests and updates their approval status. The admin provides the final approval or rejection and manages overall system resources. The central process, the Department Resource Management System, handles all these interactions, including request submission, verification, approval, and notifications.

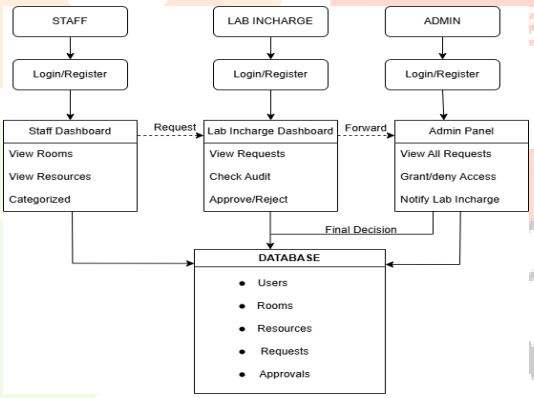

figure 2: detailed process flow of the department resource management system

The detailed process flow, illustrated in Figure 1, shows the sequential steps of the system. The process begins with Staff Login & Request Submission, where a staff member logs in, selects resources, and submits a request to the system. This is followed by Request Validation, where the system checks the request for completeness and stores it in the centralized database. Next is Lab In-charge Verification, where the Lab In-charge reviews the request, checks resource availability, and either approves or rejects it. If approved, the request is forwarded to the Admin for Final Approval/Rejection. The admin evaluates the request and updates the database with the final decision. Finally, the Notification Generation process sends automated notifications to the Staff member about the request status, and the Database Management process ensures all changes in resource availability, request logs, and user details are updated in real time.

4. SYSTEM REQUIREMENTS

4.1 Hardware Requirements the Department Resource Management System requires specific hardware to ensure smooth operation, fast processing, and reliable storage. Appropriate hardware setup is essential for supporting multiple users and real-time data management.

Server Requirements: A dedicated server with at least 4–8 GB RAM, multi-core processor, and 500 GB storage is recommended. This ensures efficient handling of concurrent user requests and centralized database management.

Client Workstations: Staff, Lab In-charge, and Admin should have computers with minimum 2 GB RAM, dual-core processor, and internet connectivity. These workstations allow seamless access to the web-based system and real-time updates.

Networking Equipment: A stable LAN or internet connection with routers and switches is required to connect all users to the server. Proper networking ensures data synchronization, fast communication, and uninterrupted workflow across departments.

4.2 Software Requirements the Department Resource Management System relies on specific software components to ensure smooth functionality, secure access, and efficient resource management. These requirements support web-based operations, database handling, and user interface interactions.

Operating System: The system can run on Windows, Linux, or macOS servers and client machines. A stable OS ensures compatibility with the Django framework and database software.

Web Development Framework: Django framework (Python-based) is required for backend development, handling request routing, authentication, and database interactions. It provides scalability, security, and easy integration with other components.

Database Management System: SQL-based databases like MySQL or PostgreSQL are used to store resource details, user data, and request logs. This ensures data consistency, reliability, and real-time updates across all modules.

Web Server: A web server such as Apache or Nginx is required to host the application and handle HTTP requests. It ensures smooth communication between client browsers and the Django backend.

Browser and Client Software:

Modern web browsers like Chrome, Firefox, or Edge are required for staff, Lab In-charge, and Admin access. They enable the web-based interface to function correctly and support interactive features.

Development Tools:

Tools like Visual Studio Code or PyCharm are needed for coding, debugging, and managing the Django application. These IDEs enhance development efficiency and maintain code quality.

Libraries and Dependencies:

Python libraries such as Django ORM, requests, and other supporting packages are required. They provide essential functionalities for database handling, web requests, and workflow automation.

5. IMPLEMENTATION

5.1 System Setup and Configuration The implementation begins with setting up the server environment, including installing the operating system, web server, and database management system. Django framework and required Python libraries are configured to support the application. User roles (Staff, Lab In-charge, Admin) are defined in the system with appropriate permissions. Resource data and user accounts are initialized in the SQL database. Network settings are optimized to ensure secure and uninterrupted access. Proper system configuration lays the foundation for smooth operation and scalability.

5.2 User Interface Development The web-based user interface is developed using HTML, CSS, and JavaScript integrated with Django templates. Staff can easily navigate to view resources, submit requests, and track request status. Lab In-charge has a dashboard to review, approve, or reject requests efficiently. Admins have a comprehensive control panel to manage resources, users, and reports. The UI is designed for usability, responsiveness, and clarity. Testing ensures that the interface works across multiple devices and browsers.

5.3 Workflow and Request Processing The workflow module is implemented to automate the flow of resource requests from Staff to Lab In-charge and finally to Admin. Each request is logged with a timestamp and stored in the database. Lab In-charge verifies availability and forwards approved requests to Admin, who makes the final decision. Notifications are automatically generated at every stage to keep users informed. This module ensures transparency, accountability, and timely resource allocation. Real-time updates prevent conflicts and errors in resource booking.

5.4 Software Description the Department Resource Management System is a web-based application developed using the Django framework with Python as the backend language. Django provides a robust structure for handling user authentication, request routing, and database operations, ensuring secure and scalable application development. The system uses an SQL-based database (such as MySQL or PostgreSQL) to store all user information, resource details, and request logs. This allows centralized storage, real-time updates, and efficient query handling for multiple concurrent users. The frontend is built with HTML, CSS, and JavaScript integrated with Django templates, providing an interactive and user-friendly interface. Staff, Lab In-charge, and Admin can easily navigate their respective dashboards to perform tasks efficiently. Additionally, Python libraries and packages like Django ORM, requests, and email handling modules are used for workflow automation, notification generation, and seamless backend operations. This combination of technologies ensures a reliable, maintainable, and fully functional resource management system.

6. CONCLUSION AND FUTURE ENHANCEMENT

6.1 Conclusion the Department Resource Management System successfully addresses the challenges of manual resource allocation by providing an automated, transparent, and efficient web-based platform. The system's role-based access model ensures secure operations for Staff, Lab In-charge, and Admin users, while the centralized SQL database maintains data consistency and real-time updates. Automated notifications and structured workflows enhance communication and accountability across departments. The implementation of Django framework and SQL database ensures scalability, security, and reliable performance. Overall, the system significantly improves resource utilization, reduces administrative workload, and supports data-driven decision-making within institutional environments.

6.2 Future Scope Future enhancements could include mobile application development for increased accessibility, integration with calendar systems for better scheduling, and implementation of AI-based resource usage analytics for predictive allocation. Additional features such as resource sharing between departments, automated maintenance scheduling, and advanced reporting capabilities could further improve system functionality. Incorporating IoT devices for real-time resource tracking and implementing blockchain technology for enhanced security are also potential areas for future development.

Acknowledgment We are deeply thankful to Almighty God for His blessings and guidance throughout the completion of this project. We are grateful to our beloved Principal Dr. R. RADHAKRISHNAN, M.E., Ph.D., Adhiyamaan College of Engineering (An Autonomous Institution), Hosur for providing the opportunity to do this work in premises. We acknowledge our heartful gratitude to Dr. G. FATHIMA, M.E., Ph.D., Professor and Head of the Department, Department of Computer Science and Engineering, Adhiyamaan College of

Engineering (An Autonomous Institution), Hosur, and the supervisor for her guidance and valuable suggestions and encouragement throughout this project and made us to complete this project successfully. We are highly indebted to Mrs. VINODHINI K M.E., Supervisor, Assistant Professor, Department of Computer Science and Engineering, Adhiyamaan College of Engineering(Autonomous), Hosur, whose immense support encouragement and valuable guidance were responsible to complete the project successfully. We also extension our thanks to Project Coordinator and all Staff Members for their support in complete this project successfully. Finally, we would like to thank to our parents, without their motivational and support would not have been possible for us to complete this project successfully.

7. References:

[1]. A. Silberschatz, H. F. Korth, and S. Sudarshan, Database System Concepts, 7th Edition, McGraw-Hill Education, 2020.

[2]. Adrian Holovaty and Jacob Kaplan-Moss, The Definitive Guide to Django: Web Development Done Right, Apress, 2nd Edition, 2019.

[3]. Jon Duckett, HTML and CSS: Design and Build Websites, John Wiley & Sons, 2011.

[4]. K. S. Trivedi and P. Varakhedi, "A Study on Resource Booking and Allocation Systems in Academic Institutions," International Journal of Computer Applications, vol. 190, no. 25, pp. 22 29, 2023.

[5]. M. Narayanan and S. Rajeswari, "Role-Based Access Control for Institutional Digital Systems," International Journal of Information Technology and Computer Science, vol. 14, no. 2, pp. 40–47, 2022.

[6]. G. S. Fathima and V. Krishnan, "Web-Based Resource Management for Educational Departments," International Journal of Emerging Technologies in Learning (IJET), vol. 17, no. 8, pp. 101–110, 2022.

[7]. Django Official Documentation, "Django Web Framework," Available at: https://docs.djangoproject.com/ (Accessed: October 2025).

[8]. Tutorials Point, "SQL Tutorial," Available at: https://www.tutorialspoint.com/sql/ (Accessed: October 2025).

[9]. W3Schools, "Web Technologies: HTML, CSS, JavaScript Tutorials," Available at: https://www.w3schools.com/ (Accessed: October 2025).

[10]. GeeksforGeeks, "Full Stack Web Development with Django," Available at: https://www.geeksforgeeks.org/ (Accessed: October 2025).

[11]. A. Kumar and P. Singh, "Automation of Departmental Inventory Using Web Technologies," International Journal of Computer Engineering Research, vol. 18, no. 3, pp. 51–59, 2023.

[12]. R. Gupta and A. Roy, "Workflow Automation for Resource Allocation in Higher Educational Institutions," International Journal of Advanced Computer Science and Applications, vol. 13, no. 9, pp. 76–84, 2023.

[13]. Stack Overflow Documentation, "Django ORM and Model Queries," Available at: https://stackoverflow.com/ (Accessed: October 2025).