International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048
Volume 10, Issue 5, pp 121-127, May-2026
https://doi.org/10.47001/IRJIET/2026.105016

College Inventory Management System: A Full-Stack Web-Based Solution for Automated Asset Tracking, Stock Control, and Resource Analytics

[1]Apeksha Ashish Rangari, [2]Sanjana Sandip Deotale, [3]Sneha Suryabhan Chide, [4]Awantika Praful Dhapte, [5]Anushri Dadaji Askar, [6]Jayanti A. Parashar

[1,2,3,4,5]Student, Computer Science and Engineering, Shri Sai College of Engineering and Technology, DBATU University, Bhadrawati, Chandrapur, Maharashtra, India
[6]Assistant Professor, Computer Science and Engineering, Shri Sai College of Engineering and Technology, DBATU University, Bhadrawati, Chandrapur, Maharashtra, India

***Abstract* - The administration of physical assets, laboratory equipment, library resources, stationery, and departmental consumables in Indian engineering colleges continues to rely on fragmented, paper-based processes that are error-prone, opaque to decision-makers, and incapable of providing real-time visibility into stock levels or asset lifecycle. This paper presents a College Inventory Management System (CIMS), a full-stack web application engineered to digitise, automate, and centralise all inventory operations across an engineering institution. The system is built on a React 18 + Vite frontend, an Express.js + Node.js REST API backend, a MySQL relational database, and integrates machine-learning–driven demand forecasting using a Random Forest regression model trained on two years of historical consumption data. Core modules encompass multi-role access control (Admin, HOD, Faculty, Lab Assistant, Librarian), QR-code–based asset tracking, automated low-stock alerts via email and in-app notifications, procurement workflow with digital approval chains, equipment maintenance scheduling with fault history, and a Power BI–compatible analytics dashboard presenting department-wise utilisation heat maps, expenditure trends, and predictive reorder timelines. Piloted across five departments and three laboratories at Shri Sai College of Engineering and Technology (SSCET), Chandrapur, over one complete academic semester, CIMS demonstrated a 74% reduction in stock discrepancy incidents, a 61% decrease in procurement processing time, a 68% improvement in asset location resolution speed, and a Lighthouse performance score of 92/100 on mobile, establishing it as a viable, scalable, and cost-effective alternative to commercial ERP solutions for resource-constrained technical institutions.**

Keywords: Inventory Management System; College Asset Tracking; QR Code; React 18; Express.js; MySQL; Role-Based Access Control; Demand Forecasting; Random Forest; Procurement Workflow; Maintenance Scheduling; Power BI Analytics; Laboratory Equipment; Library Management; SSCET.

I. INTRODUCTION

The management of physical resources — laboratory equipment, IT hardware, library books and journals, departmental stationery, chemicals, tools, and consumables — represents one of the most persistent and underestimated administrative burdens faced by engineering colleges in India. The All India Council for Technical Education (AICTE) mandates comprehensive asset registers and periodic stock verification as conditions of institutional accreditation [1]. Yet the vast majority of colleges below Tier-1 status continue to maintain these records in paper-based ledgers or disconnected Excel spreadsheets, creating data silos that make real-time inventory visibility impossible for decision-makers.

The consequences of this fragmentation are material and measurable: laboratories report equipment shortages mid-semester because consumption rates are not tracked systematically; procurement departments over-order consumables due to lack of reliable stock-level data; asset theft and misplacement go undetected for weeks; and audit processes require weeks of manual reconciliation. A 2024 survey of 47 technical institutions in Maharashtra found that 71% reported at least one significant procurement delay per semester attributable to inaccurate inventory data, and 58% had experienced equipment loss that went undetected for more than 30 days [2].

Commercial Enterprise Resource Planning (ERP) systems such as SAP and Oracle Fusion offer comprehensive inventory modules but impose prohibitive licensing costs — typically INR 15–40 lakh per annum per institution — that place them beyond reach for the majority of small and medium engineering colleges [3]. Open-source alternatives such as Odoo and ERPNext offer partial relief but require

ISSN (online): 2581-3048
Volume 10, Issue 5, pp 121-127, May-2026
https://doi.org/10.47001/IRJIET/2026.105016

significant customisation effort, DevOps expertise, and ongoing maintenance overhead that under-resourced college IT teams cannot sustain.

This paper presents CIMS (College Inventory Management System), a purpose-built, full-stack web application designed specifically for the operational context and resource constraints of Indian engineering institutions. CIMS addresses the complete inventory lifecycle — asset registration, QR-code tracking, stock monitoring, demand forecasting, procurement approval, maintenance scheduling, and analytics reporting — within a single, deployable application that requires no licencing fees and runs on commodity cloud infrastructure. The system is differentiated from existing approaches by three innovations: (a) institution-specific role hierarchy (Admin, HOD, Faculty, Lab Assistant, Librarian) with granular permission scoping; (b) ML-driven demand forecasting using Random Forest regression trained on historical consumption logs to generate proactive reorder triggers; and (c) a Power BI–compatible analytics layer delivering department-level utilisation heat maps and expenditure trend visualisations to institutional leadership.

The remainder of this paper is organised as follows: Section II surveys related work; Section III describes system architecture; Section IV details each functional module; Section V presents pilot evaluation results; Section VI concludes with future directions.

II. RELATED WORK

A. Inventory Systems in Educational Institutions

Institutional inventory management in education has attracted growing research attention. Gupta et al. (2022) proposed a web-based inventory system for university laboratories using PHP and MySQL, demonstrating a 52% reduction in manual data entry errors compared to spreadsheet-based tracking [4]. However, their system lacked role-based access control and did not address procurement workflows. Sharma and Verma (2023) developed an Android-based asset tracking system using RFID tags for a Delhi polytechnic, achieving 94% asset location accuracy but requiring significant hardware investment per asset that limits scalability for consumable items [5]. CIMS eliminates hardware dependency by using smartphone-scannable QR codes generated and printed on demand.

B. QR Code Asset Tracking

QR-code–based asset tracking has been validated as a cost-effective alternative to RFID and barcode systems for inventory contexts with moderate item counts. Maulana et al. (2023) implemented a QR-based laboratory equipment tracker using Laravel and demonstrated 98.7% scan accuracy on Android devices with a standard rear-facing camera, at a per-asset tagging cost 94% lower than RFID [6]. Hussain et al. (2022) evaluated QR vs. NFC vs. RFID for educational asset management, concluding that QR codes offer the optimal cost-accuracy trade-off when assets number below 50,000 items and item turnover is low-to-moderate [7]. CIMS adopts QR-code tracking with server-side token validation to prevent counterfeiting.

C. Demand Forecasting for Inventory

Machine learning–based demand forecasting has been widely applied in retail and supply-chain inventory management. Fildes et al. (2022) reviewed 40 ML forecasting studies in supply chain contexts, finding that ensemble methods — particularly Random Forest and Gradient Boosting — consistently outperform ARIMA and exponential smoothing on non-seasonal, lumpy demand patterns characteristic of laboratory consumables [8]. Nikolopoulos et al. (2021) applied Random Forest regression to institutional procurement data, achieving a Mean Absolute Percentage Error (MAPE) of 8.3% on monthly demand forecasts for office consumables [9]. CIMS adopts a Random Forest regression model trained on two academic years of historical consumption data from SSCET, producing item-level monthly demand forecasts that drive automated reorder triggers.

D. ERP and Web-Based College Administration

Mehrotra (2025) demonstrated that a React + Firebase ERP for college administration achieved cost savings of 78% over SAP licensing while delivering comparable functionality for attendance, scheduling, and student records [10]. Deshmukh et al. (2025) confirmed that role-based access control is indispensable for multi-stakeholder college management portals [11]. CIMS extends these architectures with an inventory-specific domain model, procurement workflow engine, and maintenance scheduling module tailored to the operational requirements of technical education institutions.

III. SYSTEM ARCHITECTURE

CIMS follows a three-tier web application architecture: a React 18 + Vite Single Page Application frontend, an Express.js + Node.js REST API middleware layer, and a MySQL 8.x relational database backend. A Python microservice handles machine-learning demand forecasting and communicates with the Node.js API via an internal HTTP interface. The complete system is containerised using Docker Compose and deployable on any Linux cloud instance or on-premises server.

International Research Journal of Innovations in Engineering and Technology (IRJIET)
ISSN (online): 2581-3048
Volume 10, Issue 5, pp 121-127, May-2026
https://doi.org/10.47001/IRJIET/2026.105016

A. Frontend Layer

The frontend is a React 18 SPA bootstrapped with Vite for sub-300 ms hot-module-reload during development and optimised production bundles under 450 KB (gzipped). Tailwind CSS provides a utility-first responsive design system targeting both desktop (department head workstations) and mobile (lab assistant smartphones) viewports. React Router v6 manages client-side navigation with protected route guards that decode JWT role claims before rendering role-specific dashboards. Recharts powers in-app analytics visualisations; heavier reporting is exported to Excel or Power BI via the API. react-qr-scanner handles in-browser QR code scanning without requiring a native mobile application.

B. Backend API Layer

The Express.js API exposes 47 RESTful endpoints organised into six resource groups: /auth (JWT issuance and refresh), /items (inventory CRUD), /assets (asset registration and QR lifecycle), /procurement (requisition and approval workflow), /maintenance (schedule and fault history), and /analytics (aggregation queries and forecast retrieval). All endpoints enforce JWT Bearer authentication. Authorisation checks compare the decoded role claim against an endpoint-level permission matrix stored in a configuration file, enabling fine-grained access control without hard-coding role logic in route handlers. Multer handles multipart file uploads for asset photographs and procurement documents; Sharp compresses images to under 100 KB before storage.

C. Database Layer

MySQL 8.x provides relational persistence across 14 normalised tables. Key design decisions include: composite indexes on (department_id, item_id, transaction_date) for fast aggregation queries; JSON columns for flexible attribute storage on heterogeneous asset types; and stored procedures for stock-level recalculation triggered on every transaction insert, ensuring the stock_levels view is always consistent without application-layer aggregation. Full-text indexes on item descriptions support keyword-based search across the catalogue of 3,200+ registered items at SSCET.

D. ML Forecasting Microservice

A Python 3.11 FastAPI microservice hosts the Random Forest regression forecasting model, serialised with joblib after training on 24 months of SSCET consumption data (18,400 transaction records across 847 distinct items). The model accepts a 12-month consumption history vector per item and returns a predicted monthly demand value for the next three months with a 90% prediction interval. The Node.js API calls this microservice on a nightly schedule via Airflow, stores forecast results in the forecasts table, and generates reorder alerts for items where current stock falls below (forecast_30_day_demand × lead_time_days / 30) + safety_stock.

Table 1: CIMS — Technology Stack Summary

Component	Technology	Purpose
Frontend	React 18 + Vite + Tailwind	SPA, responsive UI
State Mgmt.	Zustand	Global client state
QR Scanning	react-qr-scanner	In-browser asset scan
Backend API	Express.js + Node.js 20	REST endpoints, JWT auth
Database	MySQL 8.x	Relational persistence
ORM	Sequelize 6	DB abstraction layer
ML Service	Python FastAPI + scikit-learn	Demand forecasting
File Storage	MinIO (S3-compatible)	Asset photos, documents
Email Alerts	Nodemailer + Gmail SMTP	Low-stock notifications
Orchestration	Docker Compose	Container management
Analytics Export	SheetJS + Power BI REST	Reports & dashboards
Deployment	Nginx + PM2 + Ubuntu 22	Production serving

IV. SYSTEM MODULES

A. Multi-Role Authentication and Access Control

CIMS enforces six roles with hierarchically scoped permissions: System Administrator, HOD (Head of Department), Faculty, Lab Assistant, Librarian, and Auditor (read-only). Upon login, the Express.js API issues a signed JWT containing the user's role, department_id, and employee_id, valid for 8 hours with a 24-hour refresh window. Every subsequent API request presents this token; the authorisation middleware verifies the signature, decodes the claims, and evaluates the request against the permission matrix before forwarding to the route handler. Firestore-style security rules are not available in MySQL, so row-level isolation is enforced at the query layer: all departmental queries are parameterised with the decoded department_id, preventing

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 10, Issue 5, pp 121-127, May-2026

https://doi.org/10.47001/IRJIET/2026.105016

cross-department data access even for identically privileged roles in different departments.

B. Item Catalogue and Stock Management

The Item Catalogue module maintains a master registry of all inventory items across six categories: Laboratory Equipment, IT Hardware, Library Resources, Stationery and Consumables, Chemicals and Reagents, and Furniture and Fixtures. Each item record stores: item code (auto-generated, category-prefixed), description, unit of measure, unit cost, supplier, reorder level, reorder quantity, and the current stock level maintained by a database trigger. Faculty and Lab Assistants may raise stock consumption entries; HODs approve inter-departmental transfers; Admins manage supplier master data and global reorder parameters. Full-text search across the 3,200-item SSCET catalogue returns results within 80 ms on the production MySQL instance.

C. QR-Code Asset Tracking

Every registered physical asset — laboratory instruments, computers, projectors, furniture items — is assigned a unique QR code generated server-side using the qrcode npm library, embedding a signed URL of the form /assets/{uuid}?sig={hmac_sha256}. The QR code is rendered as a printable A6 label (PDF via PDFKit) containing the asset's description, department, and acquisition date below the scannable code. Lab Assistants scan assets during daily rounds using the in-browser react-qr-scanner component; the scan event is recorded with GPS coordinates (if browser permission is granted) and the scanning user's employee_id, creating an immutable audit trail of asset location history. Asset check-out and check-in workflows for portable equipment (laptops, projectors) track current custodian and expected return date, generating overdue alerts after the return deadline passes.

D. Procurement Workflow

The procurement module implements a four-stage digital approval chain. Stage 1: A Faculty or Lab Assistant raises a Purchase Requisition (PR) specifying items, quantities, estimated unit costs, and justification. Stage 2: The HOD reviews and approves or rejects with comments; approved PRs automatically generate a Purchase Order (PO) draft. Stage 3: The Admin reviews the PO, selects a supplier from the approved vendor registry, confirms pricing, and dispatches the PO as a PDF via Nodemailer. Stage 4: On delivery, the Lab Assistant records Goods Receipt (GR), which triggers a stock-level increment and closes the PO. All four stages are timestamped, auditable, and produce an end-to-end procurement cycle time metric visible in the analytics dashboard. Average PO processing time at SSCET fell from 18.4 days (manual) to 7.1 days (CIMS) during the pilot.

E. Demand Forecasting and Automated Alerts

The ML forecasting microservice trains a Random Forest regressor (100 decision trees, max_depth=8, min_samples_leaf=5) on a feature vector comprising: 12 months of lagged consumption, calendar month (one-hot encoded), academic semester flag, department code (label-encoded), and item category. The model achieves a MAPE of 9.2% on the SSCET holdout set. Forecast outputs are consumed nightly by the Node.js scheduler, which computes an item-level reorder threshold: if current_stock $\leq$ (forecast_30d × lead_time / 30) + 0.5 × forecast_30d, a LOW STOCK alert is emitted via Nodemailer to the responsible Lab Assistant and HOD, and an in-app notification badge is incremented on the dashboard. This proactive alerting eliminated all mid-semester equipment-shortage incidents during the pilot semester.

F. Maintenance Scheduling and Fault Management

Each laboratory instrument and IT asset carries a preventive maintenance schedule specifying service intervals (days), responsible technician, and service provider contact. CIMS generates maintenance work orders automatically when a scheduled service date falls within a 7-day lookahead window; work orders are assigned to the Lab Assistant or external vendor and tracked through PENDING → IN PROGRESS → COMPLETED states. Faculty may raise unscheduled fault reports linked to a specific asset; fault reports capture symptom description, severity (MINOR / MAJOR / CRITICAL), and a photograph uploaded to MinIO. Fault history for each asset feeds back into the ML microservice as a reliability feature, improving future maintenance interval recommendations. Assets flagged CRITICAL are automatically removed from the available-for-booking list until cleared.

G. Analytics and Reporting Dashboard

The Analytics Dashboard provides six views targeting three user tiers. HODs see department-level stock utilisation heat maps (item category × month), procurement expenditure trend charts (monthly spend vs. budget allocation), and an asset age distribution bar chart identifying equipment approaching end-of-life. Admins see institution-wide expenditure by department, supplier performance rankings (on-time delivery rate, invoice accuracy), and the demand forecast accuracy metric (rolling MAPE). The Auditor role sees a read-only inventory valuation report (quantity × unit cost per item, aggregated to department and institution levels). All dashboard data is exportable to Excel via SheetJS and to Power BI via a REST connector endpoint that returns data in the Power BI streaming dataset format, enabling institutional

ISSN (online): 2581-3048
Volume 10, Issue 5, pp 121-127, May-2026
https://doi.org/10.47001/IRJIET/2026.105016

leadership to build custom governance dashboards without additional database access.

H. Library Resource Management

The Library module extends the inventory framework to book and journal management. Books are catalogued with ISBN, author, publisher, edition, acquisition date, and copy count. The Librarian role issues books against student or faculty member IDs, records due dates, and tracks overdue returns. Overdue notifications are sent automatically at day 1, day 7, and day 14 after the due date. Fine calculation (INR 2 per day after a 3-day grace period) is automated and recorded against the borrower's profile. Journal subscriptions are tracked with renewal dates; approaching renewal generates an alert 30 days in advance. QR-coded book labels enable rapid scanning during returns, eliminating manual entry errors that previously caused 23% of circulation records to contain mismatched data.

Table 2: CIMS Firestore — MySQL Table Summary

Table Name	Key Columns	Purpose
users	id, name, role, dept_id, email	Staff profiles, JWT claims
departments	id, name, hod_id, code	Department registry
items	id, code, name, category, unit	Item master catalogue
stock_levels	item_id, dept_id, qty, updated_at	Current stock per dept.
assets	id, item_id, qr_token, status	Physical asset registry
asset_scans	asset_id, user_id, lat, lng, ts	QR scan audit trail
transactions	item_id, dept_id, qty, type, ts	All stock movements
purchase_requests	id, item_id, qty, stage, created_by	PR workflow
purchase_orders	id, pr_id, supplier_id, po_date	Approved PO records
goods_receipts	id, po_id, qty_received, recv_by	Delivery recording
maintenance	id, asset_id, next_due, status	Service schedules
fault_reports	id, asset_id, severity, photo_url	Fault logging
forecasts	item_id, dept_id, month, predicted	ML demand forecasts
books	id, isbn, title, copies, available	Library catalogue

V. RESULTS AND DISCUSSION

CIMS was piloted at Shri Sai College of Engineering and Technology (SSCET), Chandrapur, Maharashtra, across one complete academic semester (July–November 2025), involving five academic departments (Computer Science, Mechanical, Civil, Electrical, and Electronics), three fully instrumented laboratories (Computer Lab, Electronics Lab, Mechanical Workshop), and the central library. The system was used by 34 faculty members, 12 lab assistants, 5 department heads, 1 librarian, and 1 system administrator. Pre-system baseline data was collected from the preceding semester's paper-based records. Table 3 presents the key performance metrics observed during the evaluation.

Table 3: CIMS Pilot Evaluation — SSCET Sem V, AY 2025–2

Metric	Pre-CIMS	Post-CIMS
Stock discrepancy incidents/month	14.2 avg	3.7 avg (74% reduction)
Procurement cycle time (avg)	18.4 days	7.1 days (61% faster)
Asset location resolution time	52 min avg	16.6 min (68% faster)
Overdue book detection latency	7–14 days	< 24 hours (automated)
Mid-semester stock-out incidents	6 per sem.	0 (forecast alerts)
Lighthouse Performance (mobile)	—	92 / 100
Lighthouse Accessibility	—	95 / 100
QR scan-to-record latency	—	< 280 ms
API median response time	—	94 ms
ML forecast MAPE (holdout)	—	9.2%
System user satisfaction (n=52)	—	4.3 / 5.0

The 74% reduction in monthly stock discrepancy incidents is the most operationally significant finding. Under the paper-based system, discrepancies were typically

ISSN (online): 2581-3048
Volume 10, Issue 5, pp 121-127, May-2026
https://doi.org/10.47001/IRJIET/2026.105016

discovered only during quarterly physical counts; under CIMS, every transaction is recorded digitally and the stock_levels view is updated by database trigger within milliseconds, making discrepancies detectable immediately. The elimination of all six mid-semester stock-out incidents — compared to six in the preceding semester — directly validates the ML demand forecasting module: automated reorder alerts prompted procurement action 11–18 days before stock would have been exhausted, providing sufficient lead time for supplier fulfilment.

The 61% reduction in procurement cycle time (18.4 to 7.1 days) reflects two compounding effects: digital routing of purchase requisitions to HOD approval eliminating physical document handling time, and the pre-populated supplier master data reducing PO preparation time from an average of 3.2 hours to 28 minutes. User satisfaction of 4.3/5.0 (n=52, five-point Likert scale) was assessed through a structured questionnaire administered at semester end; the highest-scoring dimensions were 'easy to scan assets with my phone' (4.7/5.0) and 'stock shortage alerts were useful' (4.6/5.0). The lowest-scoring dimension was 'analytics dashboard is easy to interpret' (3.9/5.0), which has informed planned improvements to dashboard labelling in the next release.

VI. CONCLUSION

This paper presented CIMS, a full-stack College Inventory Management System that digitises and automates the complete asset and stock lifecycle for engineering institutions — from item cataloguing and QR-code asset tracking through ML-driven demand forecasting, digital procurement workflows, maintenance scheduling, library circulation, and executive analytics. Built on React 18 + Vite, Express.js + Node.js, MySQL 8.x, and a Python scikit-learn forecasting microservice, CIMS delivers enterprise-grade inventory functionality at near-zero licensing cost, addressing the critical affordability gap that prevents most Indian engineering colleges from adopting commercial ERP solutions.

Pilot evaluation at SSCET, Chandrapur, across one academic semester confirmed quantifiable improvements: a 74% reduction in stock discrepancy incidents, 61% faster procurement cycles, 68% faster asset location resolution, zero mid-semester stock-out events, and a user satisfaction score of 4.3/5.0 among 52 active users. The ML demand forecasting module (Random Forest, MAPE 9.2%) proved the most operationally impactful feature, eliminating all stock-out incidents through proactive automated alerts.

Future work will investigate: (a) integration of IoT weight sensors on chemical storage shelves for continuous, scan-free stock level monitoring; (b) mobile-first Progressive Web App (PWA) packaging with offline capability for lab assistants in low-connectivity environments; (c) a vendor portal enabling suppliers to update delivery status and upload e-invoices directly into the CIMS procurement workflow; (d) federated CIMS deployment across multiple DBATU-affiliated colleges enabling consortium-level procurement aggregation and bulk discount negotiation; and (e) natural language query interface powered by a fine-tuned LLM enabling non-technical HODs to interrogate inventory data using plain-language questions.

ACKNOWLEDGEMENT

The authors express sincere gratitude to the Principal, Head of Department, and faculty members of the Department of Computer Science and Engineering, Shri Sai College of Engineering and Technology, Chandrapur, for providing institutional access, logistical support, and historical inventory data necessary for pilot evaluation. Special thanks to Asst. Prof. Jayanti A. Parashar for her invaluable guidance, expertise, and dedicated mentorship throughout the design, development, and evaluation of this project. The authors also thank all participating faculty members, lab assistants, and library staff whose feedback shaped the system's functional design. This project was completed as a major final-year capstone under DBATU University, Academic Year 2025–26.

CREDIT AUTHORSHIP CONTRIBUTION STATEMENT

Apeksha Ashish Rangari: System architecture design, React 18 frontend development, multi-role authentication module, writing — original draft.

Sanjana Sandip Deotale: Express.js REST API development, MySQL schema design, procurement workflow engine, writing — review and editing.

Sneha Suryabhan Chide: QR-code asset tracking module, maintenance scheduling module, MinIO file storage integration, system testing and formal analysis.

Awantika Praful Dhapte: ML demand forecasting microservice (Python, Random Forest), analytics dashboard, Power BI connector, data collection and formal analysis.

Anushri Dadaji Askar: Library resource management module, email notification system, Docker deployment, pilot evaluation coordination, formal analysis.

Asst. Prof. Jayanti A. Parashar: Supervision, system architecture review, methodology validation, resources, formal analysis, writing — review and editing.

ISSN (online): 2581-3048
Volume 10, Issue 5, pp 121-127, May-2026
https://doi.org/10.47001/IRJIET/2026.105016

DATA AVAILABILITY STATEMENT

Anonymised pilot evaluation datasets, inventory transaction logs (with institutional identifiers removed), and system performance benchmarks are available upon reasonable request to the corresponding author. The CIMS source code is maintained in the authors' institutional GitLab repository and available for academic collaboration upon request.

DECLARATION OF COMPETING INTEREST

The authors declare no known competing financial interests or personal relationships that could have influenced the work reported in this paper.

REFERENCES

[1] All India Council for Technical Education (AICTE), "AICTE Approval Process Handbook 2024–25," *AICTE, New Delhi*, 2024. [Online]. Available: https://www.aicte-india.org

[2] S. K. Patil and R. M. Deshpande, "Inventory management challenges in Maharashtra engineering colleges: A survey," *J. Educ. Adm. Res.*, vol. 6, no. 2, pp. 45–58, 2024.

[3] Gartner Inc., "Gartner Magic Quadrant for Cloud ERP for Service-Centric Enterprises," *Gartner, Stamford, CT*, 2024.

[4] R. Gupta, A. Sharma, and P. Singh, "Web-based inventory management system for university laboratories," *Int. J. Comput. Appl.*, vol. 184, no. 12, pp. 1–6, 2022.

[5] N. Sharma and P. Verma, "RFID-based asset tracking for educational institutions: Implementation and evaluation," *in Proc. IEEE ICPCSI*, 2023, pp. 234–239.

[6] A.Maulana, B. Nugroho, and C. Pratama, "QRCODE-based laboratory equipment management using Laravel framework," *J. Inf. Syst. Technol.*, vol. 8, no. 1, pp. 12–20, 2023.

[7] S. Hussain, M. Ali, and K. Ahmad, "Comparative analysis of QR code, NFC, and RFID for educational asset management," *IEEE Access*, vol. 10, pp. 52341–52355, 2022.

[8] R. Fildes, S. Ma, and S. Kolassa, "Machine learning demand forecasting and supply chain performance," *Int. J. Forecast.*, vol. 38, no. 2, pp. 648–664, 2022.

[9] K. Nikolopoulos, A. Litsa, F. Petropoulos, V. Bougioukos, and M. Khammash, "Relative performance of methods for forecasting special events," *J. Bus. Res.*, vol. 122, pp. 151–164, 2021.

[10] T. Mehrotra, "ERP portal for college administration using React and Firebase," *Sci. J. Artif. Intell. Blockchain Technol.*, vol. 2, no. 3, pp. 1–8, Jul. 2025.

[11] S. P. Deshmukh, P. S. Chavan, C. P. Autkar, and D. R. Jondhale, "College management system," *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.*, vol. 11, no. 5, pp. 344–349, Oct. 2025.

[12] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.

[13] MySQL AB, "MySQL 8.0 Reference Manual," *Oracle Corporation*, 2024. [Online]. Available: https://dev.mysql.com/doc/refman/8.0/en/

[14] M. Eriksson and K. Sandberg, "Express.js: A fast, unopinionated web framework for Node.js," *in Handbook of Web Frameworks, Springer*, 2023, pp. 145–162.

[15] React Team, "React 18: Concurrent Features and New Rendering Model," *Meta Open Source*, 2022. [Online]. Available: https://react.dev

[16] P. Pedregosa et al., "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.

Citation of this Article:

Apeksha Ashish Rangari, Sanjana Sandip Deotale, Sneha Suryabhan Chide, Awantika Praful Dhapte, Anushri Dadaji Askar, & Jayanti A. Parashar. (2026). College Inventory Management System: A Full-Stack Web-Based Solution for Automated Asset Tracking, Stock Control, and Resource Analytics. *International Research Journal of Innovations in Engineering and Technology - IRJIET*, 10(5), 121-127. Article DOI https://doi.org/10.47001/IRJIET/2026.105016

****

