Stationery Inventory Management System(SIMS)

Dhiraj Kailas Ade
Dept. of Artificial Intelligence
G.H. Raisoni College Of Engineering
Nagpur,India
dhirajade754@gmail.com

Dr. Kotadi Chinnaiah
Dept. of Artificial Intelligence
G.H. Raisoni College Of Engineering
Nagpur,India
kotadi.chinnaiah@raisoni.net

Prof. Madhuri Sahu
Dept. of Artificial Intelligence
G.H. Raisoni College Of Engineering
Nagpur,India
madhuri.sahu@raisoni.net

Gunjan Dinesh Rakhunde
Dept. of Artificial Intelligence
G.H. Raisoni College Of Engineering
Nagpur,India
gunjan.071003@gmail.com

Garima Raju Somkuwar
Dept. of Artificial Intelligence
G.H. Raisoni College Of Engineering
Nagpur,India
garima12somkuwar@gmail.com

***Abstract*—Inventory management plays a crucial role in ensuring operational efficiency and profitability, particularly in retail domains dealing with small, fast-moving consumer products. Traditional inventory tracking and sales systems often suffer from manual errors, lack of real-time updates, and poor integration with financial reporting tools. To address these challenges, this research presents the development of a robust and intelligent Stationery Inventory Management System (SIMS), tailored for small-scale businesses with a focus on female-oriented products such as earrings, bindis, and body sprays.**

The objective of SIMS is to provide a comprehensive and user-friendly platform that streamlines stock management, automates GST-compliant billing, and delivers actionable financial insights. The system leverages real-time inventory tracking, itemized sales processing, and profit/loss analysis to enhance business decision-making. Additionally, SIMS incorporates a secure, role-based user management system to maintain data integrity and ensure authorized access.

Key features such as automated low-stock alerts, advanced search and filter options, customizable tax configurations, and QR/barcode integration are implemented to minimize manual efforts and improve operational accuracy. By integrating alerts and notifications for critical events and offering detailed reporting capabilities, SIMS not only reduces the chances of transactional errors but also optimizes day-to-day workflows.

In addition to the core system, this paper proposes a Level 1 prototype of an AI-based chatbot interface to be displayed on a digital screen at the shop entrance. This intelligent assistant will enable customers to inquire about product availability before entering the store, enhancing user experience and reducing in-store congestion. While this AI component is currently in the research and design phase, its partial implementation is planned as an extension to the existing system, contributing to the growing integration of artificial intelligence in small business retail management.

The development and deployment of SIMS demonstrate a scalable solution that enhances inventory visibility, supports growth, and simplifies financial oversight—ultimately contributing to the overall efficiency and success of modern retail businesses.

INTRODUCTION

Inventory management is a cornerstone of efficient business operations, especially in the retail sector dealing with diverse consumer products. Inaccurate stock tracking, slow billing processes, and poor financial reporting can significantly hinder business growth. This research presents the development of the Stationery Inventory Management System (SIMS), designed to streamline inventory tracking, automate sales processing, and enhance financial reporting for small-scale businesses dealing with female-oriented stationery products like earrings, bindis, and body sprays.

SIMS aims to provide a comprehensive, user-friendly solution for real-time stock management, GST-compliant billing, and detailed profit/loss analysis, thereby improving decision-making. The system is built using modern technologies to ensure a responsive, efficient, and scalable platform. The **frontend** is developed with **React.js** for building dynamic and interactive UIs, along with **Tailwind CSS** for rapid, responsive design, and **Axios** for seamless API communication. The **backend** leverages **Node.js** with **Express.js**, ensuring fast, non-blocking processing and efficient API routing, perfect for handling real-time transactions and updates.

For database management, the system uses **MySQL** to manage structured data such as product details, stock levels, and transaction history. **MongoDB** is integrated for handling non-structured data, providing flexibility for unstructured customer feedback, reviews, and other dynamic records.

As a forward-looking enhancement, the paper also proposes the integration of an **AI-powered chatbot interface** that will be displayed on a digital screen at the store entrance. This chatbot will help customers check product availability in real-time, improving their in-store experience and decision-making process. Although this feature is currently in the research and prototype stage, it represents the project's commitment to incorporating emerging technologies into retail management.

The SIMS project demonstrates a scalable and efficient solution that addresses the unique challenges of small businesses while laying the foundation for future innovations in AI-powered retail systems.

In the model development, we focused on four phase which includes,

1. Literature Review
2. Architecture
3. Methodology
4. Experimental Result

I. LITERATURE REVIEW

Researcher Name	Key Contributions/Technologies Used	Limitations/Future Research Directions
Chan Chin Wei et al.	Used PHP for backend, HTML/CSS/JavaScript for frontend; applied RAD methodology for iterative development.	Lacked IoT integration and support for barcode/QR scanning. Suggested future improvements in usability and scalability.
Pasaribu	Technologies used: HTML, CSS, JavaScript, PHP, MySQL for backend. Focused on reducing errors and enhancing process effectiveness.	The system was efficient but lacked advanced reporting and analysis features, which could be the focus of future research to enhance business intelligence.
Pasaribu	Highlighted IoT, RFID, barcode scanning; explored IMS integration with ERP systems and business processes.	Identified challenges with complex multi-location integrations and high costs; recommended future research on industry-specific IMS systems with enhanced flexibility.
Adeboje et al.	Developed with Android Studio for frontend and Firebase for backend, offering real-time inventory tracking and scalability	Lacked a discussion on data security challenges; no comparison with similar systems; future work could focus on security enhancements and cross-industry applicability.
Llabres	Combined Apache, MySQL, XAMPP for backend with a user-friendly GUI for efficient data management and real-time.	Focused on auto parts; scalability and integration with ERP systems were not fully explored. Future work suggested integration with advanced analytics and predictive tools.
Zhang & Pan	Leveraged MySQL for accurate and instantaneous warehouse operations tracking, enhancing data-driven decision-making	Challenges included initial setup costs and network infrastructure requirements; future research on system scalability and expanding IoT and machine learning integration is recommended.
Pawar & Patil	C# for frontend, SQL Server for backend; focused on improving accuracy and integrating billing with inventory management	Did not integrate predictive analytics or AI for trend analysis. Suggested further research on intelligent system features like automation and predictive modeling.
Prabakaran et al.	Not specified.	Calibration of sensors and data security were key challenges. Future research suggested in improving security protocols and expanding the system's scalability to larger environments.
Srivastava et al.	Not Specified	The system was standalone, with no AI or IoT integration; future work could focus on using advanced technologies for dynamic and predictive inventory management.

II. ARCHITECTURE

2.1 System Architecture:

The architecture of the SIMS (Stationery Inventory Management System) represents a modern full-stack application design that leverages a combination of powerful web technologies. The **frontend** is developed using **React.js** for building a responsive and interactive user interface, while **Tailwind CSS** is used for efficient and consistent styling. Communication between the frontend and backend is handled by **Axios**, which enables asynchronous HTTP requests. On the server side, the **backend** is built using **Node.js** and **Express.js**, exposing RESTful APIs to manage application logic. To ensure secure access, **JWT (JSON Web Token)** authentication is implemented for token-based user verification during login and API interactions.

The system uses a hybrid database architecture for optimized data handling. MySQL is responsible for managing structured data such as users, orders, payments, and sales history, ensuring strong data consistency. In parallel, MongoDB handles unstructured or fast-access data like logs, product catalogs, notifications, and search history, offering high-performance queries. The overall data flow begins with the frontend sending API requests, which are authenticated and processed by the backend. The backend then interacts with the appropriate database, depending on the data type, ensuring both secure data management and a smooth, efficient user experience.

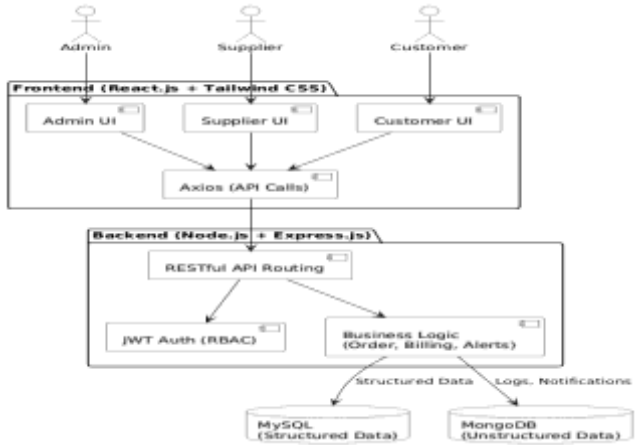

Fig1. System Architecture

2.2 Functional Architecture Working :

The second architecture diagram illustrates the operational workflow and user interaction model within the SIMS (Stationery Inventory Management System). The frontend is divided into three user interfaces based on roles: Admin UI, Supplier UI, and Customer UI. The Admin UI provides full access to system functionalities, including managing user roles, inventory control, and report oversight. The Supplier UI enables suppliers to update stock and manage product supply-related interactions. Meanwhile, the Customer UI allows customers to browse the product catalog, place orders, and complete billing processes, ensuring a streamlined user experience tailored to each role.

On the backend, core functionalities are implemented to support these roles efficiently. Authentication and authorization ensure secure access based on user roles. Alerts and notifications keep users informed in real time about inventory levels and order updates. Report generation provides admins with inventory and sales reports as needed. The system also supports inventory updates, billing, and sales processing, covering the full cycle from stock management to order fulfilment. The database layer stores and manages critical data—user roles for access control, reports for admin insights, and inventory and sales data for operational accuracy. A robust backup and security mechanism protects sensitive information and ensures data integrity throughout the system.

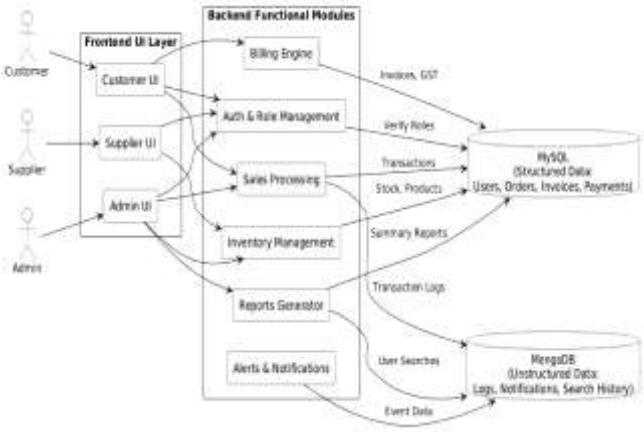

Fig2. Functional Architecture

III. METHODOLOGY

Full-Stack Development Methodology:

The SIMS application follows a full-stack development methodology, which integrates both client-side and server-side technologies to deliver a comprehensive and efficient user experience. The frontend is developed using **React.js**, a popular JavaScript library for building interactive user interfaces. React's component-based structure allows for modular development and reusability of UI elements, ensuring a consistent design and behaviour across different views and user roles (Admin, Supplier, Customer). In combination with **Tailwind CSS**, the system achieves a modern, responsive, and visually cohesive design without heavy styling overhead.

On the server side, the application utilizes **Node.js** with **Express.js** to handle all backend processes, such as API routing, authentication, business logic, and database interactions. Node.js offers an event-driven, non-blocking I/O model, making it ideal for real-time applications and scalable architectures. Express.js, a minimal and flexible Node.js framework, simplifies the setup of middleware, routing, and server-side operations, enabling rapid backend development and easy API integration.

The full-stack methodology bridges the gap between frontend and backend by allowing seamless data exchange and UI responsiveness. For example, customer actions like placing an order or admin tasks like updating stock are reflected in real time, supported by the tight integration of frontend interfaces and backend APIs. This unified development approach reduces the complexity of managing separate codebases and fosters a streamlined deployment process, allowing quicker updates and maintenance cycles.

Ultimately, this methodology ensures that the SIMS project remains robust, scalable, and maintainable. It provides end-to-end control over both user interactions and backend logic, allowing the development team to optimize the system from both technical and functional perspectives.

RESTful API Communication:

RESTful APIs (Representational State Transfer) form the backbone of communication between the frontend and backend in the SIMS architecture. By using REST principles, the system ensures that each component of the application remains loosely coupled, scalable, and independently updatable. RESTful APIs rely on standard HTTP methods—GET, POST, PUT, DELETE—to enable CRUD (Create, Read, Update, Delete) operations. These operations are mapped to specific resources, such as products, orders, inventory, and user data, allowing efficient data handling and improved modularity.

In the context of SIMS, the React.js frontend uses **Axios** to make asynchronous HTTP requests to these RESTful endpoints. For instance, when a customer places an order or an admin checks the sales report, the frontend sends a RESTful request to the backend, which processes the logic, interacts with the database, and sends a structured JSON response. This architecture enables seamless real-time interactions without the need to reload the entire webpage, resulting in a smoother and more dynamic user experience.

RESTful APIs also support **scalability and interoperability**. Because they adhere to a standard protocol, other systems or third-party services (e.g., payment gateways or inventory suppliers) can be integrated easily in the future. This flexibility

ensures that SIMS can grow as business needs evolve without having to overhaul the core infrastructure. Moreover, RESTful architecture promotes security and better debugging since each API endpoint can be individually authenticated, monitored, and tested.

Token-Based Authentication (JWT) :

Security is a critical component in any inventory and sales system, and SIMS addresses this requirement using JWT (JSON Web Token)–based authentication. Token-based authentication is a stateless, scalable solution for verifying users across different roles. When a user logs in—whether as an Admin, Supplier, or Customer—the server verifies the credentials and generates a signed JWT, which is then returned to the client and stored securely (typically in memory or local/session storage).

The token contains encrypted information such as the user ID, role, and expiry time. For every subsequent API request, the client includes this token in the header. The backend then validates the token to determine whether the request is authentic and authorized. This approach significantly reduces the need for server-side session storage, making the system lightweight and scalable. Additionally, by encoding user roles within the token, the system enforces role-based access control (RBAC), ensuring that users can only access features relevant to their roles.

JWT also enhances system security and session management. Tokens have a limited lifespan and can be refreshed or invalidated if needed, preventing unauthorized reuse. The SIMS application can restrict access to sensitive actions such as modifying inventory or generating reports, protecting the system from unauthorized manipulation or data leakage. Any attempt to access a restricted route without a valid token is rejected at the backend level, further solidifying the system's security posture.

By leveraging JWT authentication, the SIMS project achieves secure, efficient, and scalable user management across its multi-role environment. It also provides a strong foundation for implementing additional security features in the future, such as multi-factor authentication or IP-based restrictions.

Hybrid Database Architecture:

Data in an inventory management system like SIMS can be both structured and unstructured. To handle this effectively, the system adopts a **hybrid database architecture**, also known as **polyglot persistence**, combining the strengths of relational and NoSQL databases. Specifically, **MySQL** is used for managing structured data such as user profiles, orders, invoices, and transaction logs. MySQL provides strong data integrity, ACID compliance, and reliable schema management, making it ideal for core business records that require consistency and relational mapping.

On the other hand, **MongoDB** is utilized for storing unstructured or semi-structured data like product catalogs, activity logs, search history, and system notifications. These types of data are often large in volume and require flexible schema designs for rapid insertion and querying. MongoDB's document-based storage model supports this requirement and enables faster access, especially for features involving user behaviour or real-time feedback.

This dual-database approach allows the SIMS application to leverage the best of both worlds. MySQL ensures that critical business data remains accurate and reliable, while MongoDB provides the speed and flexibility needed for high-performance features. For example, an admin can generate an invoice using data from MySQL, while customers receive instant notifications powered by MongoDB.

IV. EXPERIMENTAL RESULT

The Stationery Inventory Management System (SIMS) was designed and implemented to improve the efficiency of tracking and managing stationery items in an organizational environment. The system was developed using simple and modern web technologies. The frontend was built using tools like React for building user interfaces, Tailwind CSS for styling, and Axios for connecting to the backend. On the backend side, we used Node.js and Express to create the server and connect it with a MongoDB database to store data.

Fig3. SIMS Welcome Screen

Fig4. Inventory Management Module

Fig 5. Customer Management Dashboard

V. FUTURE SCOPE

Although the current version of the Stationery Inventory Management System (SIMS) covers the basic inventory functionalities, there is significant potential for future enhancement. One of the major improvements would be developing a mobile application version of SIMS, allowing

users to manage inventory on the go and improving accessibility. Additionally, the system can be expanded to handle other types of inventory or management tasks, such as lab equipment, library materials, or office assets, making it a multi-purpose tool.

Improving the user interface (UI) for better user experience and visual appeal is also part of the future scope. A more modern and intuitive design would help users navigate the system more easily. Lastly, adding a dedicated stock management module, including features like automatic low stock alerts, reorder tracking, and detailed stock reporting, would make the system completer and more suitable for real-world use in schools, colleges, and offices.

VI. CONCLUSION

The Stationery Inventory Management System (SIMS) was developed to simplify and digitize the process of managing stationery items in an organized and efficient way. Through the use of modern web technologies, the system allows users to perform essential inventory operations such as adding, updating, deleting, searching, and filtering items. A clean and responsive user interface, combined with a dynamic dashboard and sidebar, enhances usability. The backend integration ensures smooth data handling and secure communication between the frontend and the database. Though still under development, key modules such as user authentication and real-time features have been successfully implemented. SIMS has shown promising results during testing and is ready for further improvements based on user needs and future goals.

References

[1]. Pawar, A. V., & Patil, N. V. (2024). Inventory and bill management system. International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal, 4(3).

[2]. Alangari, S., & Khan, N. A. (2021). Artificially Intelligent Warehouse Management System. Asian Journal of Basic Science & Research, 3(3), 16-24. https://ajbsr.net/

[3]. Alim, I. F. H., & Isnanto, R. R. (2023). Design of inventory management information system using periodic review method. E3S Web of Conferences, ICENIS 2023. https://doi.org/10.1051/e3sconf/202344802010

[4]. Chukomin, A., Hummel, V., Hvozdov, S., Izmailova, T., & Schuhmacher, J. (2023). Development of an IoT-based inventory management solution and training module using smart bins. in Proceedings of the 13th Conference on Learning Factories (CLF 2023).

[17]. Valery F. Lukinykh, Yulia V. Lukinykh, Algorithm For The Procurement And Inventory Management In The Distribution Supply Chain, 15th international scientific conference Business Logistics in Modern Management October 15, 2015 - Osijek, Croatia

[18]. Ummi Kalsom Hassan, Shahreen Kasim, Rohayanti Hassan, Hairulnizam Mahdin, Azizul Azhar Ramli, Mohd Farhan Md Fudzee, Mohamad Aizi Salamat, Most Stationery Inventory Management System, ISSN: 2590-4043 (online).

[5]. Daftar band, A., Shaikh, M., & Waghmode, B. B. (2022). Stock Management Using IoT. International Journal of Research Publication and Reviews, 3(6), 3657-3663. ISSN 2582-7421

[6]. Ge, X. (2024). Intelligent file management system using RFID technology and improved AICT algorithm. PeerJ Computer Science. https://doi.org/10.7717/peerj-cs.1794

[7]. Gupta, R., Ashish, Yadav, A. (2022). Inventory management system. International Journal of Creative Research Thoughts (IJCRT), 10(4). ISSN: 2320-2882.

[8]. Gupta, S., Sharma, P., & Mehta, R. (2021). IoT-Based Real-Time Inventory Tracking and Management. IEEE Access, 9, 43812-43822.

[9]. Hamdy, W., Mostafa, N., & Alawady, H. (2020). An intelligent warehouse management system using

the Internet of Things. The Egyptian International Journal of Engineering Sciences and Technology, 32, 59-65. Retrieved from https://eijest.journals.ekb.eg/

[10]. Hemalatha, G., Akshay, A., Premkumar, L., Mukhunthan, M., & Raj, V. S. (2024). Intelligent inventory management system. International Research Journal of Engineering and Technology (IRJET), 11(05), 18. [11]. Iwasokun, G. B., & Alimi, S. A. (2022). Genetic Algorithm Model for Stock Management and Control. International Journal of Strategic Decision Sciences (IJSDS), 13(1), 1-20.

[12]. Jangale, V., Sahare, P., Barapatre, K., Shahare, P., & Sarmokaddam, G. (2024). IOT based smart shelve inventory management system. Journal of Emerging Technologies and Innovative Research (JETIR), 11(4), Article 382. Retrieved from https://www.jetir.org

[13]. Feng Yang, Study on Model of Supply Chain Inventory Management Based on System Dynamics, 2009 International Conference on Information Technology and Computer Science

[14]. Eren Erman Ozguven, Kaan Ozbay, A secure and efficient inventory management system for disasters, www.elsevier.com/locate/trc

[15]. S Pasaribu, J. (2021). Development of a Web Based Inventory Information System. International Journal of Engineering, Science and Information Technology, 1(2). https://doi.org/10.52088/ijesty.v1i2.51.

[16]. Srivastava, K., Kumar Choubey, D., & Kumar, J. (2020). Implementation of Inventory Management System.

[18]. Eren Erman Ozguven, Kaan Ozbay, A secure and efficient inventory management system for disasters, www.elsevier.com/locate/trc

[19]. Punam Khobragade, Roshni Selokar, Rina Maraskolhe, Prof.Manjusha Talmale, Research paper on Inventory management system, www.irjet.net (e-ISSN: 2395-0056)

[20]. Darya Plinere, Arkady Borisov, Case Study on Inventory Management Improvement, Information Technology and Management Science, Academia.edu