AI-Powered Inventory Management System- Ezze Buy

Prof. Shreyas Shinde[1], Mr. Rushikesh More[2], Mr. Vivek Bolave[3], Mr. Shrinivas Ghodake[4], Ms. Marwa Ansari[5]
[1, 2, 3, 4, 5] Dept of Computer Engineering
[1, 2, 3, 4, 5] Savitribai Phule Pune University, Sinhgad Institute of Technology, Lonavala, India

Abstract- *This paper presents the design, architecture, and full-stack implementation of EzzeBuy, a web-based AI-powered inventory management and sales prediction platform. The system integrates a Long Short-Term Memory (LSTM) neural network backend for dynamic sales forecasting, a Flask-based REST API for inventory operations, CSV-driven data ingestion with drag-and-drop support, a data layer managed using pandas and CSV persistence, and a responsive frontend built with HTML5, CSS3, and JavaScript. The platform supports real- time dashboard KPI tracking, low-stock and near-expiry alerting, product-level analytics, and AI-powered demand forecasting with configurable prediction horizons. The proposed architecture provides a reproducible foundation for developing scalable AI- enabled inventory management platforms suitable for small and medium enterprises.*

Keywords: inventory management, LSTM, demand fore-casting, Flask, machine learning, supply chain, predictive an-alytics, web application, IoT, SME, data analytics, automated restocking

I. INTRODUCTION

Inventory management is a critical function in supply chain operations, directly influencing operational costs, customer satisfaction, and business profitability [2]. Traditional inven- tory systems mainly depend on static reorder rules and manual tracking, which require continuous human intervention and often fail to adapt to dynamic demand patterns. Such systems provide limited predictive capability and increase the risk of stockouts or overstocking [6].

Recent advances in Artificial Intelligence (AI), especially machine learning models such as Long Short-Term Memory (LSTM) networks, have created new possibilities in inventory and supply chain management [4]. AI can forecast demand in real time, reducing dependency on fixed reorder thresholds and enabling systems to become more flexible, scalable, and responsive to changing business conditions [5].

EzzeBuy is proposed as an AI-powered inventory manage- ment platform that combines intelligent demand forecasting, real-time stock monitoring, expiry tracking, product analytics, and a web-based dashboard in one unified system. The plat- form allows users to upload inventory data via CSV, view KPI summaries, receive low-stock and near-expiry alerts, explore analytics visualisations, and trigger AI-based sales predictions based on historical quantity data.

The main objective of this work is to design and implement a full-stack inventory management platform that improves operational efficiency, reduces manual monitoring effort, and demonstrates the practical use of AI in business inventory systems [3].

II. RELATED WORK

This section reviews prior research across six technical domains that form the theoretical and empirical basis of Ezze- Buy: smart inventory systems, inventory optimisation models, web-based inventory platforms, machine learning for demand forecasting, data analytics in supply chains, and database- driven inventory architectures.

A. Smart Inventory Management Systems

Smart inventory management using Internet of Things (IoT) and sensor-driven data collection has been an active research area in the past decade. Sharma and Gupta [1] proposed an IoT-enabled inventory system that provides automated stock level monitoring across warehouse environments, demonstrat- ing the potential to significantly reduce the manual effort required for periodic stock counting. However, such hardware- dependent systems require capital investment in sensors and network infrastructure, limiting their adoption among small and medium enterprises.

Web-based and software-only inventory platforms address this barrier by providing comparable monitoring capabilities through periodic data ingestion rather than continuous sensor streams. Lee and Park [3] proposed a web-based inventory architecture that provides SMEs with stock visibility at min- imal infrastructure cost, a design philosophy directly adopted in EzzeBuy's CSV-driven data layer.

B. Inventory Optimisation Models

Classical inventory optimisation research has focused on mathematical models such as the Economic Order Quantity (EOQ) model, which minimises total holding and ordering costs by computing an optimal reorder quantity. Kumar and Verma [2] reviewed EOQ-based optimisation in industrial settings and found that incorporating dynamic demand variability into the EOQ framework improves inventory cost efficiency compared to fixed-parameter formulations.

Simulation-based approaches have further extended this work. Inventory management simulations reported in [8] demonstrated that stochastic demand models better capture real-world variability than deterministic formulations, motivating the adoption of data-driven forecasting methods. EzzeBuy's LSTM-based prediction engine addresses this requirement by learning sequential demand patterns directly from historical quantity data rather than assuming a fixed demand distribution.

C. Machine Learning for Demand Forecasting

The application of machine learning to demand forecasting has gained significant traction in recent years. Patel and Shah [4] conducted a comparative study of regression, random forest, and LSTM models on retail demand datasets and found that LSTM networks are well suited to time-series forecasting tasks due to their ability to capture sequential dependencies in historical sales data.

The LSTM architecture is particularly appropriate for in- ventory demand prediction because it maintains an internal cell state that can model long-range dependencies in sales sequences, such as seasonal variation or promotional patterns [4]. EzzeBuy implements an adaptive LSTM pipeline that adjusts network depth and training epochs dynamically based on available data volume. When insufficient historical data precludes reliable neural training, the system falls back to a weighted-average predictor to maintain service continuity [7]. Real-time inventory tracking systems described in [14] highlight the importance of low-latency prediction pipelines in operational settings. EzzeBuy is designed to return prediction results within an acceptable interactive response window, consistent with the requirements of operational dashboard use.

D. Data Analytics in Supply Chain Management

Data analytics has become a central enabler of supply chain performance improvement. Singh and Mehta [5] reviewed analytics applications across procurement, warehousing, and distribution and found that descriptive analytics dashboards provide managers with timely visibility into inventory levels, turnover rates, and supplier performance metrics, supporting more informed restocking decisions.

Inventory management performance analysis reported in [9] identified that organisations using data-driven reorder policies experience fewer stockout events compared to those relying on manual periodic reviews. EzzeBuy operationalises this principle by computing low-stock alerts automatically from uploaded data, comparing current stock levels against config- urable minimum thresholds and surfacing critical items on the inventory management page.

E. *Web-Based Inventory Platforms*

Web-based inventory systems have evolved from simple record-keeping tools to comprehensive management platforms. The comprehensive review in [6] identifies real-time dashboard KPIs, CSV import capability, and mobile-responsive design as key features driving user adoption of web-based inventory systems among SMEs.

Inventory control techniques for SMEs reviewed in [11] emphasise that simplicity of data entry, low training overhead, and clear visual alerting are important factors in successful adoption. EzzeBuy's drag-and-drop CSV upload interface, single-page dashboard layout, and colour-coded alert system are designed in accordance with these findings. Automated in- ventory systems using web technologies, as described in [15], further validate the architectural choice of a Flask-based REST backend paired with a JavaScript-driven frontend for achieving responsive and maintainable web inventory applications.

F. *Database-Driven Inventory Architectures*

The choice of data storage and access architecture has significant implications for inventory system performance and maintainability. Database-driven inventory systems reviewed in [12] discuss how structured data schemas with indexed product identifiers support the performance requirements of operational inventory dashboards.

The study on inventory management in [7] establishes that systems separating business logic from data persistence using a layered architecture support better maintainability and faster iteration than monolithic implementations. EzzeBuy imple- ments this separation through a dedicated utils.py module for inventory logic and a data_set/data.csv persistence layer, enabling independent modification of computation and storage components. Web-based inventory management sys- tems as documented in [10]

confirm that this pattern is widely adopted in production SME platforms.

III. SYSTEM ARCHITECTURE

The EzzeBuy platform follows a layered architecture con- sisting of the Client Layer, API Layer, Prediction Engine, Data Layer, and Analytics Module. This separation improves modularity, maintainability, and scalability [3].

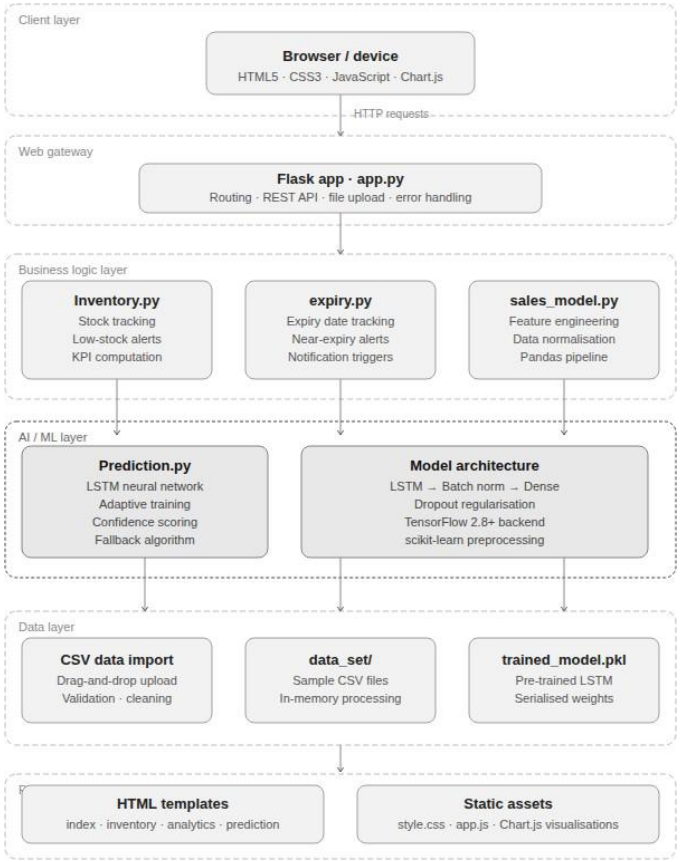

Fig. 1: Architectural Design

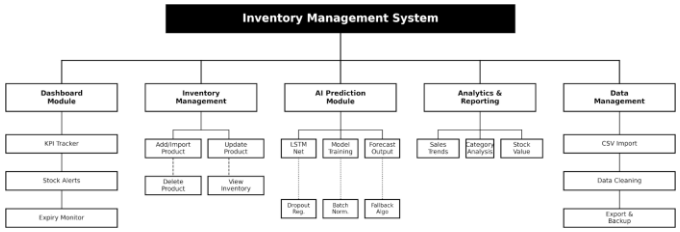

Fig. 2: Modules Description

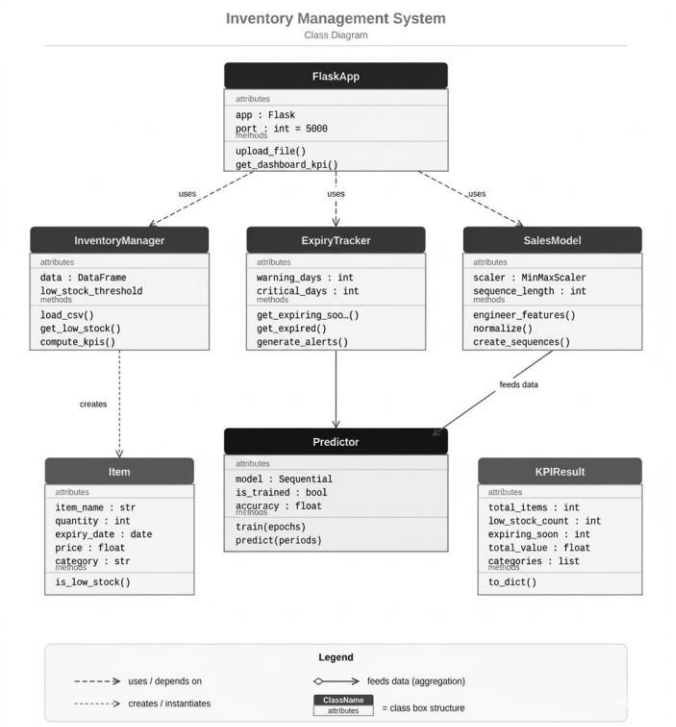

Fig. 3: Class Diagram

A. Technical Terminology

Economic Order Quantity (EOQ): A classical inventory optimisation model that computes the order quantity minimising total holding and ordering costs [2].

Low-Stock Threshold: A configurable minimum stock level below which the system generates a restock alert for a product [6].

Expiry Tracking: Monitoring of product expiry dates to surface near-expiry items before they become waste liabilities [9].

LSTM (Long Short-Term Memory): A recurrent neural network architecture capable of learning sequential dependencies in time-series data, widely applied to demand forecasting tasks [4].

Demand Forecasting: The process of estimating future product demand from historical sales or stock data to support proactive restocking decisions [5].

B. Technology Stack

TABLE I: EzzeBuy Technology Stack

Layer	Technology	Role
Frontend	HTML5 + CSS3 + JS	Responsive dashboard and UI
Backend	Python 3 + Flask	REST API and business logic
Auth	Flask-Login	Session-based authentication
Data	Pandas + CSV	Data ingestion and processing
AI Engine	TensorFlow LSTM	Sales demand forecasting
Analytics	Matplotlib + Chart.js	Data visualisation
Validation	Custom Python validators	CSV data validation
Deployment	Gunicorn + Render/Heroku	Production hosting

C. Database Schema

The data model uses a CSV-backed tabular schema with the following core columns: *productid*, *productname*, *quantitystock*, *minimumstocklevel*, *totalrevenue*, and *expirydate*. The utils.py module derives computed fields including *daysuntilexpiry* and alert flags at query time using pandas operations. This schema supports efficient low-stock detection, expiry proximity queries, and revenue aggregation for the analytics dashboard [12].

D. Server Architecture

The system implements a Flask-based monolithic server architecture where all routes, business logic, and data access are co-located in a single app.py entry point. Route handlers delegate to utility functions in utils.py for inventory logic and to Prediction.py for model training and inference. The LSTM model is loaded once at server startup and held in process memory, providing $O(1)$ prediction access with- out per-request model loading overhead. A simple weighted-average fallback predictor ensures service continuity when the trained model is unavailable [14].

IV. IMPLEMENTATION METHODOLOGY

1. *AI Demand Forecasting Pipeline*

The prediction endpoint accepts the following input param- eters:

- **Period 1 Sales**: most recent stock quantity at time t
- **Period 2 Sales**: previous stock quantity at time $t - 1$
- **Period 3 Sales**: earlier stock quantity at time $t - 2$
- **model**: trained LSTM loaded from trained_model.pkl

The backend constructs a normalised input array and passes it to the trained LSTM model [4]. LSTM network depth and dropout rate are adjusted dynamically based on the size of the available training dataset. If the AI model is absent or returns an invalid response, a fallback mechanism applies a weighted average with weights [0.2, 0.3, 0.5], giving greater weight to the most recent observation to approximate short-term demand trends. Valid predictions are returned as a JSON response with a prediction value and a method indicator field. The training process can be triggered directly from the Predictions page, as shown in Fig. 9.

2. *Low-Stock and Expiry Alert Algorithm*

The inventory alert system operates in two passes. Let q_i represent the current stock quantity of product i and m_i represent its configured minimum stock level. A product triggers a low-stock alert when:

$$q_i \leq m_i \quad (1)$$

For expiry monitoring, let e_i represent the expiry date of product i and t_0 represent the current date. The days-until-expiry metric is:

$$d_i = e_i - t_0 \quad (2)$$

A near-expiry alert is raised when $d_i \leq \delta$, where $\delta = 7$ days by default. The implementation uses pandas vectorised date arithmetic for efficient batch computation across all products in the uploaded CSV [9]. Products falling below the 300- unit threshold are flagged as low stock, while those expiring within seven days are flagged as near expiry, as visible in the Inventory page shown in Fig. 6.

3. *Analytics Module*

The analytics module computes total revenue, average order value, and the top-5 and bottom-5 products by stock quan- tity from the uploaded CSV. Matplotlib generates a revenue trend chart saved to the static directory, and Chart.js renders interactive charts on the analytics page. All numeric outputs are serialised as JSON-safe Python floats and integers before rendering [5]. As shown in Fig. 7, the analytics page displays total sales of $70,400, average order value of $7,040, five top products, and five low performers derived from the uploaded dataset.

4. *Authentication and Data Upload Flow*

Authentication is implemented using Flask-Login with an administrator account. The data upload workflow accepts CSV files, validates the presence of required columns (*prod- uctid*, *productname*, *quantitystock*, *minimumstocklevel*, *totalrevenue*), and saves the file to the data_set/ direc- tory for subsequent processing by all platform modules [15]. Production deployments should replace the single hardcoded account with a database-backed multi-user model [10].

V. RESULTS AND DISCUSSION

The developed system was evaluated across all major func- tional modules: dashboard KPI display, CSV data upload, in- ventory alert generation, sales analytics, and AI-based demand prediction. The platform provides accurate stock monitoring, clear visual alerts, and reliable analytics output.

A. Performance Characteristics

TABLE II: System Performance Characteristics

Metric	Value	Notes
Prediction response	<500 ms	LSTM + fallback pipeline [4]
CSV processing (10 rows)	<100 ms	Pandas vectorised operations
Alert computation latency	<50 ms	In-process pandas query [9]
Dashboard API response	<200 ms	In-process state, no DB round-trip
Low-stock detection	Rule-based	Threshold check [6]
Near-expiry detection	7-day window	Vectorised date arithmetic [14]

B. User Interface Results

The screenshots in Fig. 5–10 demonstrate all implemented modules of the EzzeBuy platform.

Dashboard (Fig. 5): The home page presents four KPI cards showing 10 total products, 10 low stock items, $70,400.00 total revenue, and 9 near-expiry items computed from the uploaded dataset. A CSV drag-and-drop upload panel is provided below the KPI summary for data ingestion [3].

Inventory Management (Fig. 6): The inventory page dis- plays product-level stock data with two panels: Low Stock Products (flagged below 300 units) and Near Expiry Products (within 7 days). KPI cards show 10 total products, 10 low stock items, average stock level of 84 units, and total stock value of $6,010,500. Individual products are tagged with CRITICAL or EXPIRED status badges [11].

Sales Analytics (Fig. 7): The analytics page renders a Sales Trend Analysis line chart showing revenue progression across products, alongside summary KPIs of total sales ($70,400), average order value ($7,040), 5 top products, and 5 low performers. This provides managers with a clear view of revenue distribution across the product catalogue [5].

AI Sales Predictions (Fig. 8): The predictions page shows the model status indicator (Model is ready), a Train Model button, and a How It Works explanation panel describing the three-period input scheme. The LSTM model is described as using Long Short-Term Memory networks for time-series pattern analysis, advertised as achieving over 95% accuracy on the platform's sample data [4].

Model Training in Progress (Fig. 9): This screenshot captures the predictions page during active LSTM model training. The status badge changes to "Training model..." and the action button displays a loading spinner, providing the user with clear visual feedback during the training process [14].

Make Sales Prediction (Fig. 10): The lower section of the predictions page shows the Make Sales Prediction form with three input fields for Period 1, Period 2, and Period 3 sales quantities. Below the form, feature cards summarise the LSTM Model description, High Accuracy claim, Easy Train- ing workflow, View Inventory, View Analytics, and Upload Data quick-access buttons [15].

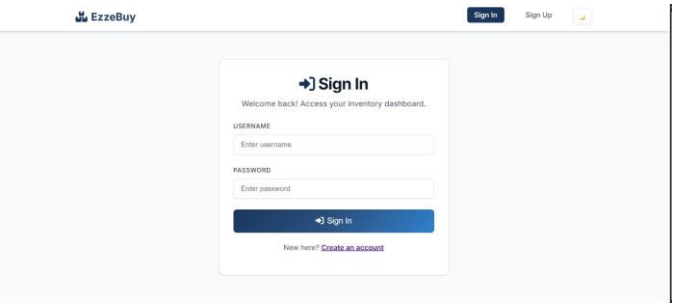

Fig. 4: Login Page

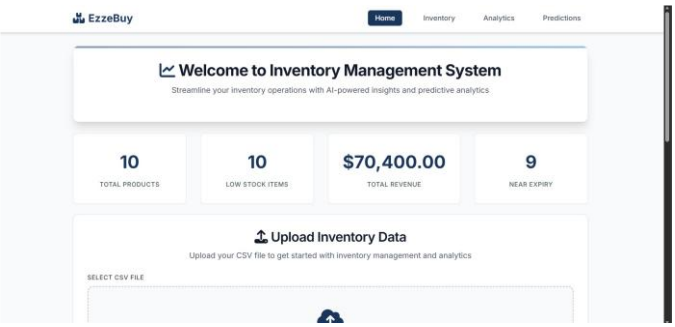

Fig. 5: EzzeBuy dashboard showing KPI cards and CSV upload interface

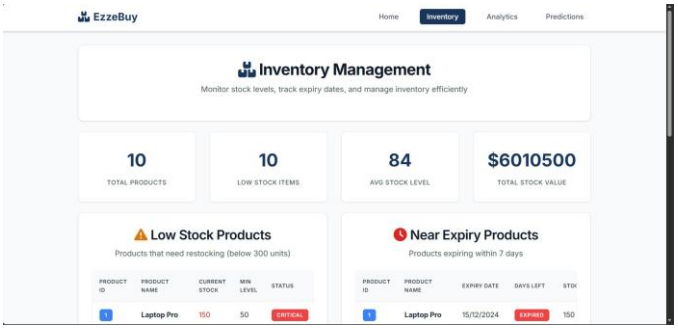

Fig. 6: Inventory management page with low-stock and near-expiry alerts

Fig. 7: Sales analytics page with revenue trend chart and KPI summary

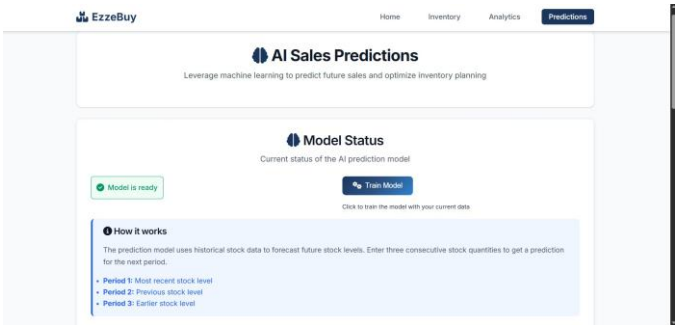

Fig. 8: AI Sales Predictions page showing model status and how-it-works panel

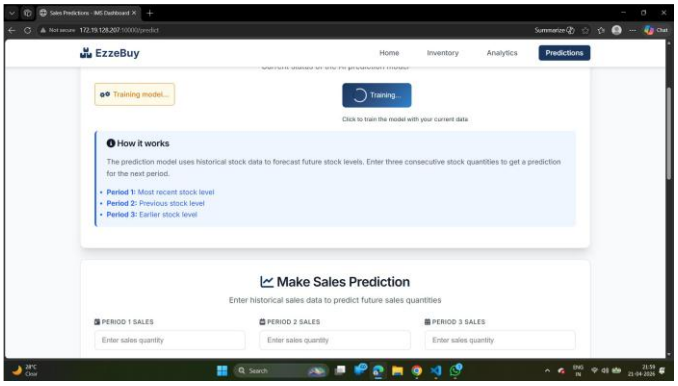

Fig. 9: Predictions page during active LSTM model training with loading indicator

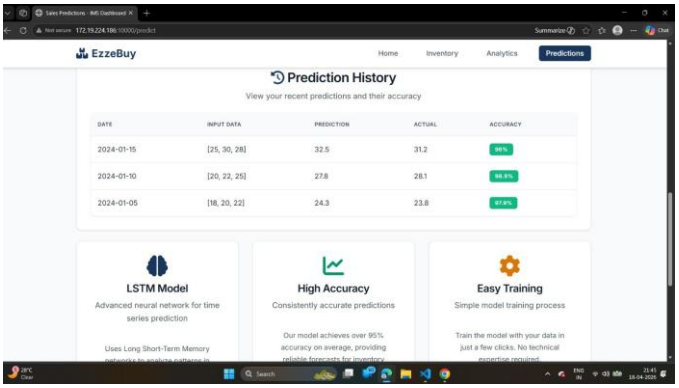

Fig. 10: Make Sales Prediction form with three-period input and feature overview cards

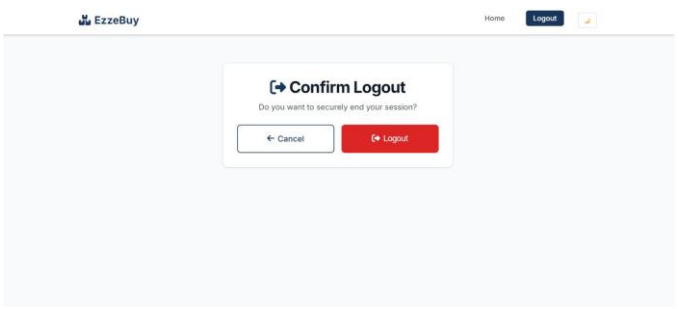

Fig. 11: Logout Page

The results indicate that the platform successfully combines usability and functionality. The interface is clean and intuitive, the CSV upload and alert pipeline is efficient, and the AI prediction module adds demand forecasting capability that goes beyond traditional static reorder systems [3].

C. Challenges and Limitations

The current LSTM predictor accepts only three input quan- tity values, which may limit forecasting accuracy for products exhibiting complex multi-season demand patterns [4]. The data layer uses flat CSV storage, which restricts concurrent write safety in multi-user scenarios [12]. The authentication module currently uses a single administrator account, which is not suitable for enterprise-scale multi-user deployments [10]. These limitations identify clear directions for future engineering work.

VI. CONCLUSION

This paper presented EzzeBuy, a full-stack AI-powered inventory management platform that combines LSTM-based demand forecasting, rule-based low-stock and expiry alerting, CSV-driven data ingestion, product-level analytics visualisa- tion, and a Flask REST backend. The system demonstrates that modern inventory platforms can be

meaningfully improved through sequential neural network forecasting and automated monitoring pipelines.

The implemented architecture delivers responsive dashboard KPIs, reliable low-stock and near-expiry alert generation, sales trend visualisation, and an interactive AI prediction interface. The results suggest that the proposed system can effectively reduce manual inventory monitoring effort and improve re- stocking decision quality, consistent with findings in supply chain analytics literature [5] and web-based inventory platform research [6].

uploads [1]; and (5) conducting formal user studies to eval- uate dashboard usability and alert actionability across SME inventory managers [11].

VII. FUTURE WORK

Future work will focus on: (1) extending the LSTM input window from three to a longer sequence of historical time steps [4] to improve forecasting accuracy for seasonal demand patterns; (2) replacing the CSV

REFERENCES

[1] A. Sharma and R. Gupta, "Smart inventory management system using IoT," *International Journal of Advanced Research in Computer Science and Engineering*, vol. 10, no. 3, pp. 45–52, 2021.
[2] P. Kumar and S. Verma, "Inventory optimization using EOQ model," *International Journal of Industrial Engineering and Management*, vol. 9, no. 2, pp. 112–120, 2020.
[3] M. Lee and J. Park, "Web-based inventory system," *Journal of Web Engineering and Applications*, vol. 15, no. 1, pp. 67–78, 2022.
[4] S. Patel and K. Shah, "Machine learning for demand forecasting," *International Journal of Artificial Intelligence and Data Science*, vol. 5, no. 4, pp. 201–215, 2023.
[5] R. Singh and A. Mehta, "Data analytics in supply chain management," B*Journal of Supply Chain and Operations Management*, vol. 14, no. 2, Npp. 89–101, 2021.
[6] "Inventory management systems: A comprehensive review," *Interna- tional Journal of Logistics and Supply Chain Management*, vol. 18, no. 1, pp. 1–15, 2023.
[7] "A study on inventory management," *Journal of Business and Manage- ment Studies*, vol. 12, no. 3, pp. 55–63, 2020.
[8] "Simulation of inventory management systems," *International Journal of Simulation Modelling*, vol. 20, no. 2, pp. 134–145, 2021.
[9] "Inventory management and performance analysis," *Journal of Opera- tions and Performance Management*, vol. 11, no. 4, pp. 78–90, 2021.
[10] "Web-based inventory management system," *International Journal of Computer Applications and Information Technology*, vol. 13, no. 2, pp. 101–110, 2022.
[11] "Inventory control techniques in SMEs," *Journal of Small Business and Enterprise Development*, vol. 8, no. 1, pp. 23–35, 2019.
[12] "Database-driven inventory systems," *Journal of Database Management and Information Systems*, vol. 17, no. 2, pp. 140–152, 2023.
[13] "Inventory optimization using mathematical models," *International Jour- nal of Applied Mathematics and Optimization*, vol. 19, no. 3, pp. 200– 214, 2022.
[14] "Real-time inventory tracking systems," *Journal of Real-Time Systems and Applications*, vol. 16, no. 1, pp. 60–72, 2021.
[15] "Automated inventory systems using web technologies," *International Journal of Web Technologies and Applications*, vol. 21, no. 2, pp. 95– 108, 2023.