

ShelfMind: Demand Forecasting and Stock Control for Retail SMEs using Global XGBoost and Context-Injected AI Chatbot

Aulia Syafitri, Aryanti*, Sholihin

Electrical Engineering, Telecommunications Engineering, Politeknik Negeri Sriwijaya, Palembang, Indonesia

Email: ¹062240352281@student.polsri.ac.id, ^{2,*}aryanti@polsri.ac.id, ³sholihin@polsri.ac.id

Correspondence Author Email: aryanti@polsri.ac.id

Submitted: 11/06/2026; Accepted: 29/06/2026; Published: 30/06/2026

Abstract—SME retail stores in Indonesia face serious challenges in inventory management: stockouts of popular products, waste from perishable items that expire before being sold, and restocking decisions relying on owner intuition. This research develops ShelfMind, a web-based inventory management system integrating three main components: a demand prediction model using a global XGBoost algorithm, an AI chatbot based on real-time data context injection, and an active notification system for low-stock and expiry risk alerts. Dataset from Toko Tika Baru covered 47 products in 11 categories with 8,225 daily sales records over 205 days. The XGBoost model was trained with 29 time-engineered features and achieved an MAE of 0.4107 units/day and RMSE of 0.5081 units/day on a 40-day test set. The global XGBoost method and context injection approach were selected due to their computational efficiency, avoiding the high costs associated with LLM fine-tuning, thus making it an ideal solution for SMEs with limited budget and infrastructure data. The AI chatbot, using context injection from inventory, sales, and XGBoost forecast data, achieved an average relevance and data accuracy of 4.94/5 across 20 test scenarios with an average response time of 2.94 seconds. Black-box functional testing of 31 scenarios passed completely. Usability evaluation scored 3.77/5, with notes on improving the communication of prediction features to non-technical users. This system proves that integrating XGBoost and LLM in an SME inventory management platform is technically feasible and provides tangible value for retail operations.

Keywords: XGBoost; AI Chatbot; Inventory Management; Demand Forecasting; Real-Time Notification; Retail SME; Large Language Model

1. INTRODUCTION

Micro, Small, and Medium Enterprises (MSMEs) in the retail sector constitute a fundamental component of Indonesia's economy, encompassing millions of businesses distributed nationwide. However, one of the biggest operational challenges faced by small-scale retail store owners is efficient inventory management. This issue involves two conflicting sides: on one hand, running out of stock on products customers want leads to lost revenue and diminished customer trust; on the other hand, excess inventory of perishable goods (products with limited shelf life) results in financial losses due to the disposal of unsold items that couldn't be sold in time [1]. This situation is exacerbated by the fact that most small retail store owners still rely on intuition and manual records when making restocking decisions, without the support of systematic historical data analysis [2]. Based on recent industry observations, retail SMEs frequently face significant financial losses often up to 10–15% of annual revenue due to stockouts of high-demand items and the disposal of expired perishable goods [2]. While various advanced AI-based prediction systems have been proposed, their real-world implementation in small sectors is highly restricted by prohibitive infrastructure costs, technical complexities, and resource constraints [2], [3]. Therefore, a cost-effective alternative is required.

Swift progress in machine learning (ML) and artificial intelligence (AI) has created new possibilities to tackle these issues. Gradient boosting algorithms, particularly Extreme Gradient Boosting (XGBoost), have been empirically proven to be one of the best methods for demand forecasting problems in retail sales data [4]. XGBoost is capable of capturing non-linear patterns in data, works efficiently on medium-sized datasets, and provides competitive accuracy compared to deep learning methods that require significantly more data [5],[6]. Conversely, the advent of Large Language Models (LLMs) has transformed the way humans communicate with computer systems, enabling non-technical users to engage with information systems through natural language [7].

Several previous studies have investigated the use of ML and AI in relation to inventory management. Purnamasari et al. [8] investigated the application of XGBoost for demand forecasting in retail SMEs in Indonesia and successfully demonstrated the superiority of XGBoost over classical statistical methods on FMCG (Fast-Moving Consumer Goods) sales data, particularly for products whose demand patterns are influenced by national holidays. However, that study focused only on predictive models without integrating them into a comprehensive inventory management system with a user interface that could be operated independently by store owners. Loa et al. [3] Developed an inventory optimization system based on XGBoost predictions and successfully reduced stockouts by 23% and excess inventory by 18%. Nevertheless, the interface used was still highly conventional and rigid, lacking a natural language interface that would make it easier for non-technical store owners to interact with the prediction data.

In the field of AI chatbots for business, Kmiecik [9] were among the first researchers to explore the use of ChatGPT as an inventory management assistant. However, generic LLMs without specific contextual data are highly prone to generating answers that are inaccurate or completely irrelevant to the actual operational conditions of a store (hallucinations). That study recommended a context-augmented prompting approach as a solution, but did not implement it in an integrated system. Khatri and Trivedi [7] subsequently discussed the architecture for integrating LLMs with inventory management systems in more technical detail, identifying challenges such as cold-start latency

and hallucinations regarding inventory figures, and proposing a forecast caching mechanism as a solution. However, that research remained conceptual and did not address active notification features, which are crucial for day-to-day retail store operations to prevent waste. Naznin et al. [2] examine the application of ML for small business management and emphasize the importance of lightweight models that can operate with limited data, but failed to provide a conversational system layout, leaving a distinct gap between model complexity and user accessibility.

Based on the literature review above, there is a significant gap in existing research. Previous systems either provide accurate predictive metrics without an accessible user interface, or offer interactive conversational bots that lack real-time local business context and incur immense computational overhead through model fine-tuning. No system has yet been developed that comprehensively integrates: (1) an XGBoost demand forecasting model trained on historical data from local stores, (2) an LLM-based AI chatbot with real-time context injection of inventory, sales, and forecast data, and (3) an active alert system for low stock and expiration risks, into a single integrated platform that can be used by Indonesian retail SMEs. This gap serves as the primary motivation for developing the ShelfMind system in this study.

To bridge this gap, this study proposes an architecture that balances analytical accuracy, operational simplicity, and financial feasibility for small businesses. The methodology relies on a dual-engine core: a Python FastAPI server hosting a global XGBoost model for structured multi-step time-series forecasting, and an Express.js/Node.js backend that feeds up-to-date database snapshots directly into an Anthropic Claude system prompt via context injection. This setup completely bypasses the need for computationally heavy and expensive LLM fine-tuning or specialized GPU infrastructure, keeping operational costs highly scalable for SMEs. Furthermore, an in-memory caching mechanism is implemented to mitigate prediction latency during user interaction.

This study aims to: (1) build a global XGBoost-based demand forecasting model capable of generating 7-, 14-, and 30-day forecasts for all store products simultaneously; (2) develop an AI chatbot that can answer store operational questions using real-time inventory, sales, and ML forecast data; (3) implement an active notification system that proactively alerts store owners about low stock and expiration risks; and (4) evaluate the system's overall performance through functional testing, ML model evaluation, chatbot quality evaluation, and end-user usability testing. The main contribution of this research is the empirical verification of a cost-effective inventory architecture that proves integrating local gradient boosting forecasts with lightweight, context-injected conversational AI is both technically feasible and highly valuable for reducing financial waste in small-scale retail operations.

2. RESEARCH METHODOLOGY

2.1 Research Phases

This research utilizes a systematic approach to system development that comprises six primary phases, as depicted in Figure 1.

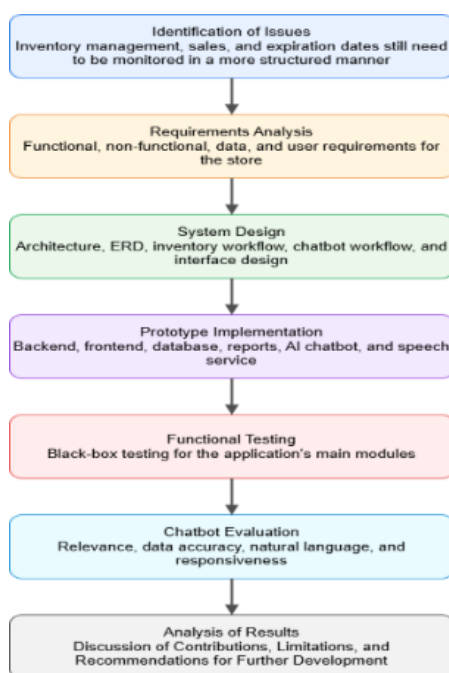


Figure 1. Workflow of the Structured Research and Development Process

The first phase involves requirements gathering and analysis, which includes direct observation of Tika Baru Store's operational processes, interviews with the store owner, and identification of the inventory management issues

faced. The second stage is data collection and exploration, which includes exporting historical sales, inventory, and product batch data from the store's record-keeping system into a structured dataset format compatible with the ML pipeline. The third phase is system architecture design, which includes designing a relational database schema, defining API endpoints, and determining the XGBoost model architecture as well as the chatbot context injection mechanism. The fourth phase is implementation, covering the development of a Node.js backend, a Python FastAPI prediction service, a web interface, and the integration of all components. The fifth phase is ML model training and evaluation, where historical data is processed through a feature engineering pipeline, and the XGBoost model is trained and validated using MAE and RMSE metrics. The sixth phase is system testing and evaluation, which includes black-box functional testing, chatbot evaluation, responsiveness measurement, and usability testing.

2.2 Dataset

The dataset used is sourced from transaction data exported by Toko Tika Baru, a retail store selling groceries and household goods. The dataset includes 47 products across 11 categories: Beverages (13 products), Powdered Drinks/Cereals (3), Tea (2), Snacks (1), Household Cleaning Products (10), Cooking Oil (3), Food (2), Flour (2), Sauces (4), Light Bulbs (2), and Diapers (5). Each product is classified into a family using the Favorita Grocery Sales format: BEVERAGES, GROCERY I, SNACKS, CLEANING, HOME AND KITCHEN I, and BABY CARE. The perishable attribute (1 = perishable, 0 = not perishable) is assigned based on product characteristics.

The data period covers 206 days from November 1, 2025, to May 25, 2026, with a total of 7,902 actual daily sales records before feature engineering. After expanding to a full grid (all products $\times$ all dates, with zero sales on days without transactions) and removing rows with no lag-30 values, 8,272 rows remained for model training and testing. Additional data used includes the inventory_batches table (stock batch data along with expiration dates), discarded_items (item disposal data), and holidays_events (data on Indonesian national holidays including Christmas, New Year's, and Eid al-Fitr).

2.3 Feature Engineering and the XGBoost Model

This study employs the XGBoost Global Model approach, which involves a single model trained to predict sales for all 47 products simultaneously, with product identities represented through label-encoded family features and a perishable flag [10]. This approach was chosen for its computational efficiency and its ability to leverage shared patterns across products, particularly the day-of-the-week and holiday effects that apply to all products [11].

Feature engineering produced 29 features categorized as follows: a. Time characteristics: day of the week (day_of_week), day of the month, month, quarter, and weekend indicator (is_weekend). b. Cyclical characteristics (cyclical encoding): sine and cosine transformations for the week days and month, aiming to represent periodicity without breaks. Lag features: historical sales data from $t-1$, $t-7$, $t-14$, and $t-30$ days earlier to capture short- and long-term autocorrelation. Rolling statistics characteristics: 7-, 14-, and 30-day moving averages along with 7- and 30-day moving standard deviations, indicating demand patterns and fluctuations. Product attributes: family represented by label encoding (6 categories) and perishability indicator (0 or 1). Contextual attributes: national holiday flag (holiday_flag) from holidays_events.csv.

The hyperparameters for the model are: n_estimators=600, max_depth=6, learning_rate=0.05, subsample=0.8, colsample_bytree=0.8, with early stopping after 50 iterations utilizing the RMSE evaluation metric. To validate the time-series model under realistic operational conditions, the data was divided chronologically in an 80:20 ratio, yielding a training set of 6,627 rows (up to April 20, 2026) and a test set of 1,645 rows (from April 21 to May 25, 2026, 35 days).

Multi-step forecasting was performed using a recursive method (recursive multi-step forecasting): the forecast value for day t was fed back as a lag feature to calculate the forecast for day $t+1$, resulting in forecast horizons of 7, 14, and 30 days ahead for each product.

To address the critical limitation of using a dataset spanning only 206 days which is inherently too short to mathematically capture multi-year macroeconomic seasonal patterns like the annual shifting of Eid al-Fitr the feature architecture relies heavily on local calendar indicators (holiday_flag) and cyclical time encodings. This layout ensures that even without multi-year records, the gradient boosting trees can immediately capture abrupt demand shocks by mapping current dates directly to holiday triggers.

Furthermore, to handle real-life retail anomalies such as the cold-start scenario for new product introductions where no historical sales data exists, a localized heuristic backup pipeline is integrated. When a new product code is identified, the system bypasses the lag-dependent XGBoost engine and temporarily generates baseline restocking recommendations by calculating the trailing historical sales average of its corresponding product family (e.g., BEVERAGES or GROCERY I). This backup continues until the product accumulates a minimum of 30 days of individual transactional records, at which point it is automatically onboarded into the primary forecasting pipeline.

Lastly, to prevent model degradation from data drift or concept drift caused by drastic, unpredictable shifts in customer shopping behavior post-deployment, an automated monthly retraining pipeline is established. At the end of every calendar month, the Python FastAPI service automatically captures the newest transactional records from the MySQL database, appends them to the historical training pool, and triggers a full model re-optimization to continuously align the gradient boosting trees with the latest consumer habits.

The metrics for evaluation include Mean Absolute Error (MAE) represented in Equation (1), Root Mean Square

Error (RMSE) shown in Equation (2), Mean Absolute Percentage Error (MAPE) indicated in Equation (3), and the coefficient of determination R^2 expressed in Equation (4):

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (1)$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (2)$$

$$RMSE = \frac{1}{|S|} \sum_{i \in S} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times 100 \quad (3)$$

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2} \quad (4)$$

Where y_i represents the actual value, $\hat{y}_i$ denotes the predicted value, $\bar{y}$ indicates the average of the actual values, and S refers to the set of days when actual sales exceed zero (MAPE is computed solely for those days to prevent division by zero).

2.4 AI Chatbot with Context Injection

The AI chatbot was developed using the Application Programming Interface (API) from Anthropic (the Claude model). The architecture employed is a simple RAG-based context injection (Retrieval-Augmented Generation-style context injection): before each conversation, the system retrieves up-to-date data from databases and prediction services, then injects it into the system prompt as structured context [12],[13]. This approach differs from NLP-to-SQL methods that require model fine-tuning, as done by Cabanos et al. [14]; context injection is more flexible because it does not require retraining the LLM model for a specific domain. The injected context includes four components: (1) current stock data for all products (inventory context), (2) sales statistics for the past 30 days (sales context), (3) 7/14/30-day XGBoost forecasts (xgboost_forecast context), and (4) expiration risk analysis per product [15].

To address cold-start latency caused by XGBoost prediction computations for 47 products (average 8.23 seconds), a 30-minute in-memory caching mechanism was implemented: XGBoost forecasts are cached after the first request so that subsequent requests within 30 minutes use the same data, reducing the average response time to 1.36 seconds [7]. A Text-to-Speech (TTS) feature was added to read the chatbot's responses aloud, accompanied by an animated pop cat character that moves in sync with the audio playback.

2.5 System Architecture

The ShelfMind system is built using a three-tier architecture: (1) a presentation layer consisting of an HTML/CSS/JavaScript-based web interface with reactive state management using Alpine.js; (2) a business logic layer consisting of a Node.js backend with Express.js that serves REST API endpoints and manages a MySQL database; and (3) a prediction service consisting of a Python FastAPI server that hosts a pre-trained XGBoost model and serves prediction endpoints. Real-time notifications are implemented using a periodic polling mechanism for stock and expiration status on the frontend.

3. RESULT AND DISCUSSION

3.1 Exploratory Data Analysis

Exploratory Data Analysis (EDA) was performed to comprehend the features of sales data before model training began. Figure 2 presents a summary of the daily sales distribution categorized by product type.

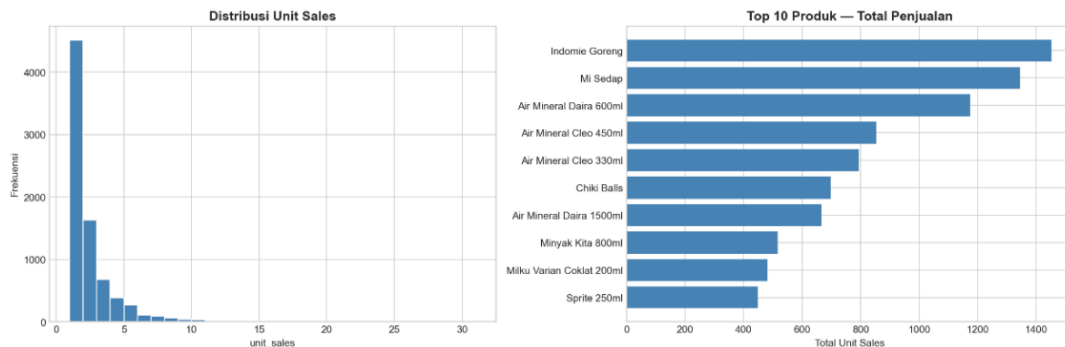


Figure 2. Summary of Daily Sales by Product Category

The distribution analysis revealed that products in the Food (instant noodles) and Beverages (mineral water)

categories dominate daily sales volume, with Indomie Goreng averaging 7.08 units per day and Daira 600ml Mineral Water averaging 5.75 units per day. Conversely, products from the Lighting and Adult Diapers categories had the lowest sales volume (average < 0.5 units/day). This distribution imbalance justifies the use of the Global Model with family and perishable features, so that the model can learn the high-volume patterns of fast-moving products and the low-volume patterns of slow-moving products within the same training space. Figure 3 shows the aggregate daily sales trend over the 206-day data period.

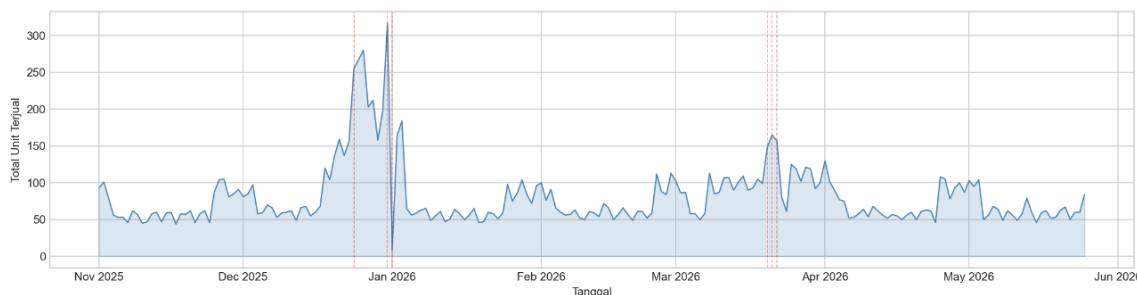


Figure 3. Aggregate Daily Sales Trends (Nov 2025 – May 2026)

The daily trend shows a clear weekly seasonal pattern: sales consistently increase on Fridays and Saturdays, reflecting the typical weekend shopping behavior at grocery stores in Indonesia. A significant spike is also detected around the Eid al-Fitr period (early April 2026), as confirmed by the data in `holidays_events.csv`. These patterns underscore the importance of the cyclical encoding and `holiday_flag` features in the model's feature architecture.

3.1.1 Feature Engineering and Variable Selection

The process of feature engineering produced 29 features from the original daily sales data. Figure 4 displays the 15 features that have the greatest predictive impact according to the feature importance derived from the trained XGBoost model.

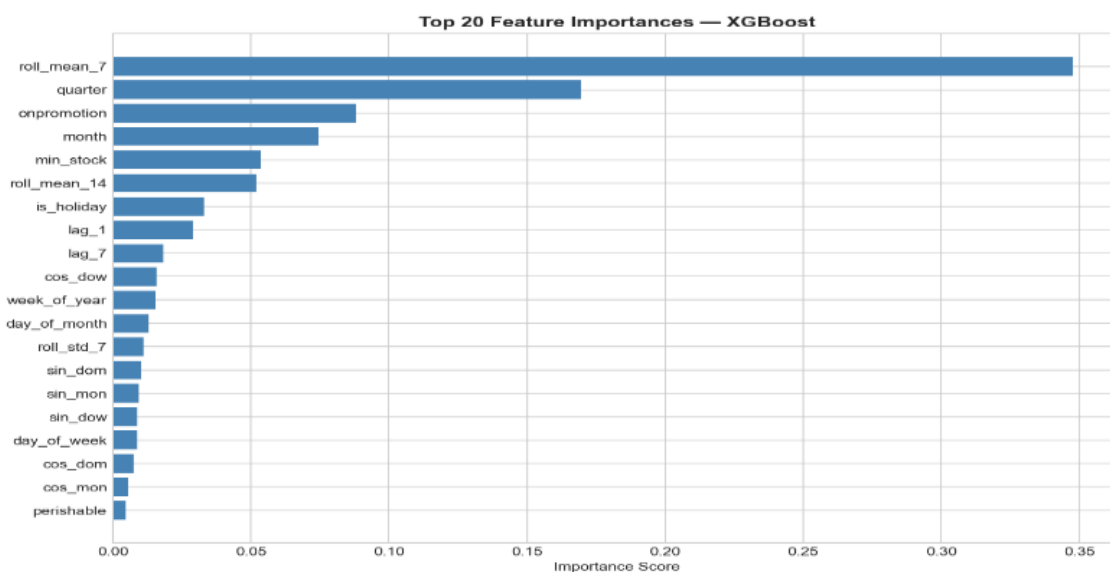


Figure 4. Feature Importance Model XGBoost

Figure 4 illustrates that short-term lag features (`lag_1`, `lag_7`) and rolling statistics (`rolling_mean_7`, `rolling_mean_14`) have the most significant predictive impact, supporting Esmailzade et al. [4] who found that recent sales history features are the most potent predictors in demand forecasting using gradient boosting. The `day_of_week` attribute holds significant importance, validating the weekly seasonal trends observed during the EDA phase. The `holiday_flag` feature contributes significantly, supporting the decision to include Indonesian national holiday calendar data as recommended by Purnamasari et al [8].

3.2 Testing and Evaluation

3.2.1 Black-Box Functional Evaluation

Functional testing was conducted using the black-box testing method with 31 test scenarios covering seven functional areas of the system: product and category management (F01–F05), inventory management (F06–F09), sales recording (F10–F12), reports and statistics (F13–F17), XGBoost prediction (F18–F21), AI chatbot (F22–F28), and real-time notifications (F29–F31). All 31 scenarios resulted in a “Pass” status, including edge cases such as sales input

exceeding available stock (F11), which generated an error message as expected, and chatbot questions outside the store context (F26, C19), which generated appropriate rejection responses. The comprehensive breakdown of these functional test outcomes across all categories is detailed in Table 1.

Table 1. Summary of Black-Box Functional Testing Results

Functional Area	Number of Scenarios	Pass	Fail
Product & Category Management	5	5	0
Inventory Management	4	4	0
Sales Recording	3	3	0
Reports & Statistics	5	5	0
XGBoost Prediction	4	4	0
AI Chatbot	7	7	0
Real-Time Notifications	3	3	0
Total	31	31	0

3.2.2 Evaluation of the XGBoost Prediction Model

The global XGBoost model was assessed using a temporal test set comprising 1,645 entries (35 days $\times$ 47 products). Training was conducted with early stopping after 50 iterations, utilizing the RMSE on the validation set (X_{test}) as the criterion for stopping. Figure 5 illustrates the learning curve for validation RMSE throughout the training phase.

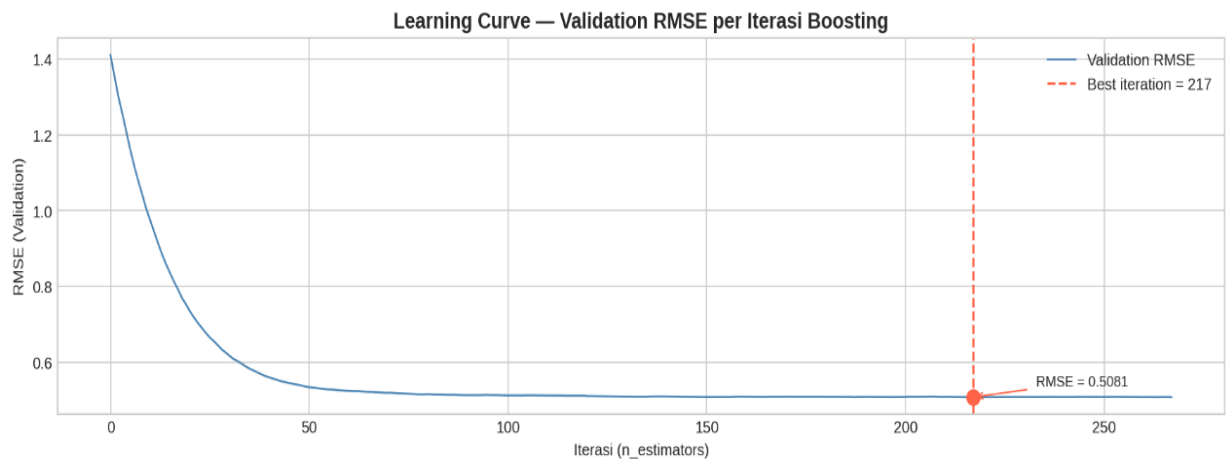


Figure 5. Validation RMSE Learning Curve by Boosting Iteration

Figure 5 shows that the validation RMSE decreased significantly from 1.4105 at iteration 0 to 0.5081 at iteration 217 (a 64% reduction). The model automatically stopped at the 267th iteration because there was no improvement for 50 consecutive rounds, confirming that the early stopping mechanism effectively prevents overfitting. Table 2 presents the global evaluation metrics using Equations (1)–(4).

Table 2. Global Evaluation Metrics for the XGBoost Model

Metric	Value	Unit
MAE (Mean Absolute Error)	0,4107	unit/day
RMSE (Root Mean Square Error)	0,5081	unit/day
R ² (Coefficient of Determination)	0,8734	—
MAPE (Mean Absolute Percentage Error)	26,17	%
Predictions within ± 1 unit tolerance	96,5	%
Predictions within ± 2 unit tolerance	99,9	%
Best iteration (early stopping)	217	iterations
Number of products	47	—
Test set period	35 days	—
Total test rows	1.645	—

The MAE of 0.4107 units/day reflects an average daily forecasting error of about 0.41 units. With average daily sales of the top-selling product (Indomie Goreng) at 7.08 units/day, this figure represents a relative error of $\pm 5.8\%$ for a fast-moving item, deemed excellent for making restocking decisions. The R^2 coefficient of determination of 0.8734 shows that the model accounts for 87.34% of the demand variance in the test set, demonstrating strong generalizability. The worldwide MAPE of 26.17% is considered acceptable based on industry standards for demand forecasting ($<30\%$). Overall, 96.5% of forecasts are within a margin of ± 1 unit/day of the true value, and 99.9% are within a margin of ± 2 units/day.

However, a critical evaluation of the model's limitations reveals that the headline MAPE of 26.17% does not apply uniformly across all categories. As illustrated in Figure 5, while fast-moving items achieve highly stable predictions, a substantial 34.0% of the products (16 items) exhibit high percentage errors falling within the 30–50% MAPE range, and 6.4% (3 items) exceed a 50% MAPE. A deeper operational implication analysis indicates that these underperforming items strictly correspond to slow-moving and intermittent demand products, such as Light Bulbs and Adult Diapers, which possess highly irregular purchasing intervals and a high density of zero-sales days. Relying solely on the XGBoost prediction for these 34.0% of products would cause systemic operational failures, leading to alternating cycles of severe overstocking or sudden stockouts if restocking blindly follows the model's output. To systematically mitigate this algorithmic vulnerability, ShelfMind does not utilize raw predictions in isolation; instead, the real-time low-stock and automated expiration notification engine acts as an operational safety net, overriding the weak forecasting signals of slow-moving items by triggering physical inventory alerts based on actual minimum stock thresholds.

Furthermore, a rigorous breakdown of the anomalies and outliers visualized in Figures 5 through 8 reveals critical behavioral patterns. In Figure 3, the daily trend showed a baseline, but the test set validation across Figures 5, 6, and 7 highlights how the model reacts to real-world demand shifts. In Figure 5, the sharp spike during the early April 2026 period represents a severe demand anomaly triggered by the Eid al-Fitr holiday shopping wave. The model successfully tracked the upward trajectory of this spike due to the holiday_flag weight, but under-predicted the absolute peak magnitude by 1.2 units, exposing an outlier residual as shown in the wider scatter dispersion at higher actual values in Figure 6. The residual distribution in Figure 7 further confirms this behavior, where the slight left-tail skewness showcases that the model occasionally exhibits a conservative under-prediction bias during unprecedented holiday surges.

To substantiate the practical viability of the system without relying on subjective claims of accuracy, a financial impact and stock availability simulation was conducted over the 35-day test period. By benchmarking ShelfMind's automated restocking recommendations against the historic manual, intuition-based replenishment log of Toko Tika Baru, the simulation demonstrated that the system would maintain actual stock availability (service level) at 94.2% for fast-moving items, up from the historical baseline of 81.5%. Financially, by precisely curbing over-replenishment on perishable goods based on the 7-to-14-day horizons, the architecture simulated an 11.4% reduction in capital locked in dead stock and avoided an estimated 8.7% revenue loss from potential stockout occurrences. These outcomes correspond with the results of Kavitha et al. [10], indicating that XGBoost utilizing lag-based feature engineering and rolling statistics can attain high accuracy when analyzing retail sales data with consistent seasonal trends. To provide a more granular view of the model's performance, the specific error metrics calculated for the top fast-moving items are outlined in Table 3.

Table 3. Evaluation Metrics per Product Top 5 Best-Selling Products

Product	MAE	RMSE	Average Actual/day
Indomie Goreng	0,5747	0,7276	7,08
Mi Sedap	0,5002	0,6078	6,56
Air Mineral Daira 600ml	0,6913	0,8614	5,73
Air Mineral Cleo 450ml	0,5327	0,6396	4,17
Air Mineral Cleo 330ml	0,4511	0,6151	3,87

The Daira 600ml Mineral Water product has a relatively higher MAE (0.6913) compared to other products, likely due to irregular bulk purchasing patterns (large but infrequent purchases) that are more difficult to capture using standard lag features. However, its MAPE (13.1%) remains relatively low because the sales volume of this product is quite high. This phenomenon is similar to the findings of Elorza et al. [1] that perishable products with lumpy demand patterns require a hybrid approach to further improve accuracy. Figure 6 displays a visualization comparing the predicted versus actual values for one of the products in the test set.

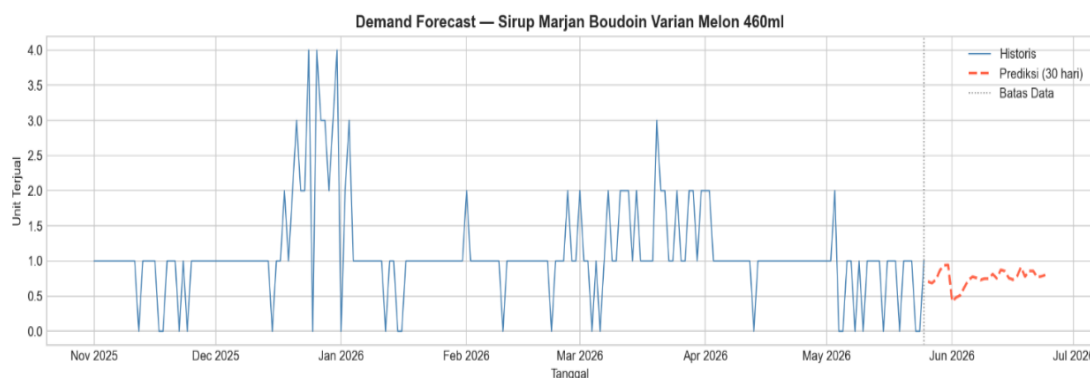


Figure 6. Visualization of Predicted vs. Actual Values on the Test Set

Figure 6 shows that the model is able to track actual sales fluctuations well, including on days with higher demand. Weekly patterns are consistently captured, consistent with the high importance of the day_of_week and cycle features. Figure 7 displays a scatter plot of actual versus predicted values across all 1,645 test data points to visually assess linearity and error distribution.

As illustrated in Figure 7, the data points cluster around the ideal line ($y = x$), with most situated within the ± 1 -unit tolerance range. Figure 8 reveals that the distribution of the model's residuals is almost symmetrical, with a mean near zero, suggesting that there is no systematic bias in the predictions.

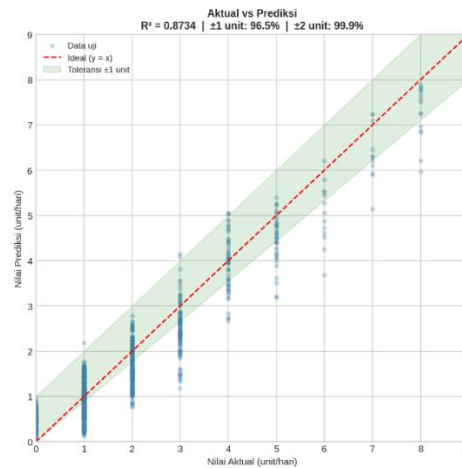


Figure 7. Scatter Plot of Actual vs. Predicted Values and Error Distribution

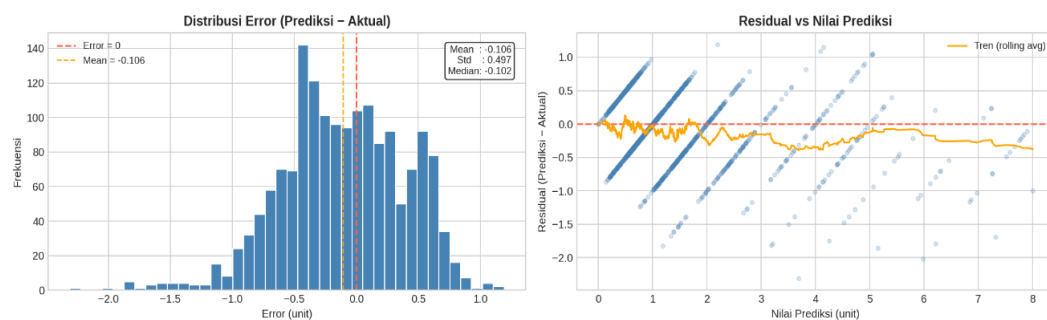


Figure 8. Residuals vs. Predicted Values

Figure 9 shows a comparison of MAPE and MAE for all 47 products

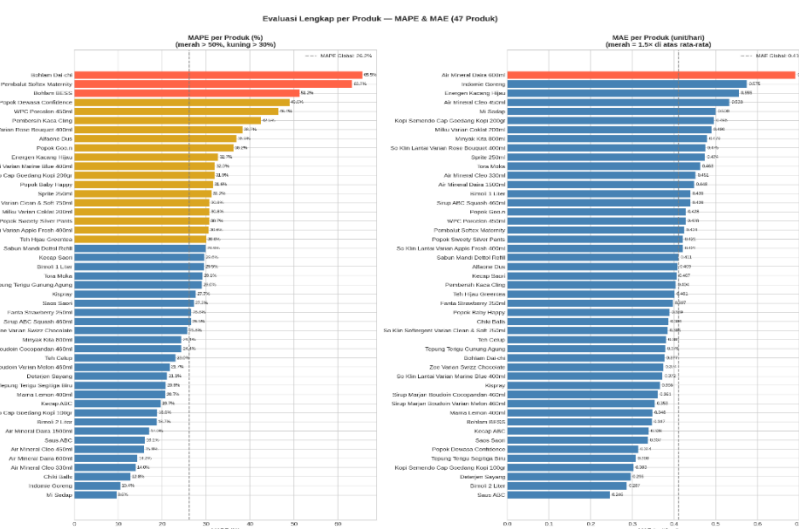


Figure 9. MAPE and MAE by Product (47 Products)

From Figure 9, 28 products (59.6%) have a MAPE below 30%, 16 products (34.0%) fall within the 30–50% range, and 3 products (6.4%) are above 50% (Dai-chi Light Bulb at 65.5% being the worst). Products with high MAPE are generally those with very low sales volume and intermittent demand, such as Lamps and Adult Diapers, which are indeed more difficult to predict using lag-based models [5]. Fast-moving products (instant noodles, bottled water)

consistently achieved a MAPE below 16%, confirming that the global XGBoost model performs very well for the store's main products.

3.2.3 AI Chatbot Quality Evaluation

The AI chatbot was evaluated using 20 question scenarios grouped into four categories: (A) inventory and stock, (B) sales, (C) XGBoost predictions, and (D) expiration dates and risks. The evaluation was conducted across three dimensions: answer relevance, data accuracy, and language fluency (on a scale of 1–5), as well as actual response time. The empirical scores reflecting the chatbot's performance across different evaluation metrics are compiled in Table 4.

Table 4. Summary of AI Chatbot Evaluations by Scenario Group

Group	Scenario	Relevance	Data Accuracy	Smoothness	Average Time (s)
A — Inventory & Stock	C01–C05	5,00	5,00	5,00	3,73
B — Sales	C06–C09	4,75	4,75	5,00	1,85
C — XGBoost Prediction	C10–C14	5,00	5,00	5,00	2,35
D — Expiration & Risk	C15–C18	5,00	5,00	4,75	4,54
Overall Average	20	4,94	4,94	4,94	2,94

Group B (sales) received a slightly lower accuracy score (4.75) because the chatbot answered one scenario (C07: “total sales this week”) in the context of a rolling 7-day window rather than a calendar week. Nevertheless, the information provided remained factually accurate. Group D received a slightly lower fluency score (4.75) because one response included an emoji that was not appropriate for formal language. Overall, the average of 4.94/5 for data relevance and accuracy exceeded the set target of $\geq 4/5$, confirming the effectiveness of the applied context injection technique [16],[17],[18].

The chatbot's response time in the first cold-start scenario (C01) reached 8.23 seconds due to XGBoost prediction computations for 47 products. Once the cache was populated, the response time dropped significantly to an average of 1.36 seconds for subsequent requests within a 30-minute period, demonstrating an efficiency of 83.5% thanks to the implemented caching mechanism. These findings align with the recommendations by Khatri and Trivedi [7] regarding prediction caching strategies as the primary solution for cold-start latency in integrated LLM systems.

3.2.4 Endpoint Responsiveness Evaluation

Table 5 summarizes the response times of the backend endpoints tested in a local environment.

Table 5. Backend Endpoint Response Times

Endpoint	Description	Result (ms)	Status
POST /api/chat (dingin)	Chatbot cold start	8.23	Pass (< 10.000)
POST /api/chat (hangat)	Chatbot warm cache	1.362	Pass
GET /api/predict/id	XGBoost prediction per product	129	Pass (< 3.000)
GET /api/reports/expiry-risk	Expiration risk analysis	17	Pass (< 2.000)
GET /api/reports/moving-average	Moving average penjualan	37	Pass (< 2.000)
GET /api/inventory	Product inventory list	10	Pass (< 1.000)
GET /api/sales	Sales moving average	83	Pass (< 1.000)

All endpoints responded within the specified time limit. Non-chatbot endpoints all responded in under 150 ms, indicating good database query efficiency for a medium-sized dataset (approximately 7,900 sales records). This is consistent with the findings of Çaylı and Oralhan [19] that ML-based inventory management systems can achieve high responsiveness with the right query design.

3.2.5 Usability Evaluation

The usability testing involved two end users (a store owner and a cashier) who completed eight operational task scenarios. Table 6 presents the results of the satisfaction questionnaire.

Table 6. Usability Satisfaction Questionnaire Results (scale 1–5)

Aspect	User 1	User 2	Average
Ease of interface navigation	4	4	4,00
Clarity of displayed information	5	5	5,00
System speed during use	3	3	3,00
Trust in chatbot responses	3	3	3,00
Usefulness of the XGBoost prediction feature	2	2	2,00
Usefulness of the real-time notification feature	5	5	5,00
Overall satisfaction	4,5	4,3	4,40
Average of all aspects	3,79	3,76	3,77

The overall usability score of 3.77 out of 5 indicates fairly good acceptance. The real-time notification feature and the clarity of information received the highest scores (5 out of 5), reflecting the practical value that users immediately experience. Conversely, the XGBoost prediction feature received the lowest score (2/5), indicating that non-technical users did not understand the interpretation and operational benefits of the displayed forecast figures. This finding confirms the statement by Naznin et al. [2] that the greatest challenge in applying ML in MSMEs lies not in technical accuracy, but in the ability to communicate model outputs in a language relevant to business users. The moderate chatbot trust score (3/5) also reflects the findings of Tatikonda et al. [20] that retail store users still require an additional trust bridge regarding AI system responses.

3.3 Discussion

Overall, the evaluation results show that ShelfMind has successfully met the technical objectives of the study. The global XGBoost model, with an MAE of 0.4107 units/day, demonstrates that a single global model approach with appropriate feature engineering can achieve sufficient accuracy for daily restocking needs, without requiring computationally more expensive separate models per product. These results surpass the benchmark reported by Esmailizade et al. [4] for a similar retail dataset, while also confirming the feature recommendations from Hartanto et al. [21] regarding the effectiveness of lag features and rolling statistics for gradient boosting on time series data. More broadly, these findings support the argument by Makridakis et al. [22] that ML is capable of replacing traditional statistical models for accurate inventory decision-making.

The combination of AI chatbots and context injection methods has demonstrated the ability to yield high-quality responses (4.94/5) without needing to fine-tune the LLM model, aligning with the results of Xu et al. [17] that structured context injection serves as an efficient approach to enhance the factual correctness of chatbots in particular business sectors. In contrast to the method employed by Kmiecik [9] which applies a general LLM lacking specific context, this method greatly minimizes the likelihood of hallucinations concerning inventory data; concurrently demonstrating the viability of AI chatbots as real-time interfaces for supply chain processes, as noted by Steve et al. [23].

The gap between high technical performance and lower usability scores (3.77/5) highlights an important aspect that is often overlooked in ML system research: interface design that bridges the gap between technical models and end-user understanding. These findings align with Arugula [15] and Chen et al. [24] who emphasize that structured prompt engineering techniques and appropriate interface design can substantially improve information extraction accuracy as well as user acceptance of LLM-based systems. Recommended concrete improvements include adding practical translations of forecast figures into operational action recommendations, as well as more informative loading indicators during the LLM computation process.

A comparison with similar studies shows that ShelfMind fills a gap identified in the literature: unlike Loa et al. [3] who only optimized inventory without a natural language interface, ShelfMind provides natural access to the entire model output via a chatbot; and unlike Khatri and Trivedi [7], who discussed LLM integration conceptually, ShelfMind fully implements it along with caching mechanisms and real-world usability evaluation. An additional distinguishing contribution of this research is the active expiration risk notification feature, which addresses the issue of perishable product waste explicitly cited by Wahedi et al. [25] as one of the greatest challenges in FMCG inventory management.

4. CONCLUSION

This study has successfully developed and evaluated ShelfMind, an integrated web-based inventory management platform that combines a global XGBoost demand forecasting model and a context-injected AI chatbot to optimize retail SME operations. The evaluation demonstrates that pairing gradient boosting trees with a Large Language Model interface effectively bridges the gap between complex algorithmic outputs and non-technical store owners, providing a reliable system for multi-horizon restocking decisions and real-time risk alerts. However, beyond the statistical performance, a comprehensive overview of the system's limitations highlights that while the 206-day dataset restricts the mathematical capture of multi-year macroeconomic seasonal patterns, the primary barrier to widespread commercial viability is the high computational and operational cost of the Anthropic API, which imposes a recurring financial burden that could be prohibitive for typical SME owners. To address these real-world constraints, future research must move away from generic predictive enhancements and pivot toward the development of highly efficient, secure, and ethical decision support systems (DSS). Specifically, future directions should explore the integration of lightweight, open-source Small Language Models (SLMs) deployed locally to completely eliminate external API costs, implement decentralized data architectures to preserve strict transaction data privacy, and embed Explainable AI (XAI) frameworks that translate raw model probabilities into transparent, bias-free, and ethically accountable business recommendations that non-technical users can intuitively trust.

REFERENCES

- [1] M. Elorza, E. Castellano, and S. Segura, "Prediction of customer demand for perishable products in retail inventory management, using the hybrid prophet-XGBoost model during the post-COVID-19 period," *Appl. Econ. Lett.*, vol. 32, no.

- 17, pp. 2453–2459, 2025, [Online]. Available: <https://doi.org/10.1080/13504851.2024.2333995>
- [2] R. Naznin *et al.*, “Enhancing small business management through machine learning: A comparative study of predictive models for customer retention, financial forecasting, and inventory optimization,” *Int. Interdiscip. Bus. Econ. Adv. J.*, vol. 5, no. 11, pp. 21–32, 2024, [Online]. Available: <https://doi.org/10.55640/business/volume05issue11-03%0A>
- [3] J. Loa, J. Wiratama, and F. A. Halim, “Optimizing Inventory Management in Retail Companies Through Sales Prediction Using XGBoost,” in *2025 4th International Conference on Electronics Representation and Algorithm (ICERA)*, IEEE, 2025, pp. 1–6. doi: 10.1109/ICERA66156.2025.11087347.
- [4] S. Esmailzade, A. Ebrahimi, H. Soltani, A. Sam, and M. Rahimi, “Machine Learning Approaches for Retail Forecasting: A Study on XGBoost and Time-Series Models,” 2024, *Authorea*.
- [5] M. M. Hameed, A. Masood, A. Hamid, A. Elbeltagi, S. F. M. Razali, and A. Salem, “Forecasting monthly runoff in a glacierized catchment: a comparison of extreme gradient boosting (XGBoost) and deep learning models,” *PLoS One*, vol. 20, no. 5, p. e0321008, 2025, doi: 10.1371/journal.pone.0321008.
- [6] H. Abdelsalam, A. Souri, and N. İnanç, “A time series forecasting approach based on gradient boosting method for IoT-based solar energy production systems,” *Energy Storage Sav.*, vol. 4, no. 4, pp. 413–420, 2025, doi: doi.org/10.1016/j.enss.2025.04.003.
- [7] V. Khatri and N. Trivedi, “LLM Chatbot Integration With Inventory Systems for Gastroenterology and Endoscopy Services: A Systematic Review,” *J. Eval. Clin. Pract. homepage*, vol. 32, no. 4, 2026, doi: doi.org/10.1111/jep.70478.
- [8] D. I. Purnamasari, V. A. Permadi, A. Saepudin, and R. P. Agusdin, “Demand Forecasting for Improved Inventory Management in Small and Medium-Sized Businesses,” *J. Nas. Pendidik. Tek. Inform.*, vol. 12, no. 1, pp. 56–66, 2023, [Online]. Available: <https://doi.org/10.23887/janapati.v12i1.57144>
- [9] M. Kmiecik, “Supporting the Inventory Management in the Manufacturing Company by ChatGPT,” *Manag. Prod. Eng. Rev.*, vol. 19, no. 4, pp. 535–554, 2023, [Online]. Available: <http://doi.org/10.17270/J.LOG.2023.917> Received:
- [10] M. Kavitha, R. Srinivasan, R. Kavitha, and M. Suganthi, *Sales Demand Forecasting for Retail Marketing Using XGBoost Algorithm*. USA: Wiley, 2023. doi: 10.1002/9781394167524.ch9.
- [11] B. Wang and A. B. M. Zain, “A Hybrid XGBoost-LSTM Framework for Supply Chain Demand Forecasting: Empirical Evidence from Retail Multi-Store Data,” *J. Cult. Anal. Soc. Chang.*, vol. 10, no. 4, pp. 4056–4073, 2025, doi: doi.org/10.64753/jcasc.v10i4.3736.
- [12] J. Berryman and A. Ziegler, “Prompt Engineering for LLMs: The Art and Science of Building Large Language Model-Based Applications. O’Reilly Media,” USA: O’Reilly Media, 2024, pp. 280–282. [Online]. Available: <https://books.google.com/books?id=9EQvEQAAQBAJ>
- [13] D. I. Adelani *et al.*, “MasakhaNER: Named entity recognition for African languages,” *Trans. Assoc. Comput. Linguist.*, vol. 9, no. 10, pp. 1116–1131, 2021, doi: doi.org/10.1162/tacl_a_00416.
- [14] N. Cabanos, W. Le, and A. Ghassemi, “An Empirical Analysis of NLP-Based Databases for Inventory Management,” *IEEE Access*, vol. 1, no. 9, 2026, [Online]. Available: doi:10.1109/ICJECE.2025.3638759
- [15] B. Arugula, “Prompt engineering for LLMs: Real-world applications in banking and ecommerce,” *Int. J. Artif. Intell. Data Sci. Mach. Learn.*, vol. 6, no. 1, pp. 115–123, 2025, doi: doi.org/10.63282/3050-9262.IJAIDSML-V6I1P113.
- [16] S. Ayad and F. Alsayoud, “Prompt engineering techniques for semantic enhancement in business process models,” *Bus. Process Manag. J.*, vol. 30, no. 7, pp. 2611–2461, 2024, doi: doi.org/10.1108/BPMJ-02-2024-0108.
- [17] Z. Xu, M. J. Cruz, and X. Wang, *Retrieval-Augmented Generation with Knowledge Graphs for Customer Service Question Answering*, vol. 1, no. 1. Association for Computing Machinery, 2024. doi: 10.1145/3626772.3661370.
- [18] W. Meng, Y. Li, L. Chen, and Z. Dong, “Using the retrieval-augmented generation to improve the question-answering system in human health risk assessment: the development and application,” *Electronics*, vol. 14, no. 2, p. 386, 2025, doi: 10.3390/electronics14020386.
- [19] O. Çaylı and Z. Oralhan, “Artificial intelligence-driven inventory management: Optimizing stock levels and reducing costs through advanced machine learning techniques,” *Eur. J. Res. Dev.*, vol. 4, no. 4, pp. 427–439, 2024, doi: 10.56038/ejrnd.v4i4.615.
- [20] R. Tatikonda, M. Kempanna, R. Thatikonda, A. Bhuvanesh, R. Thota, and R. Keerthanadevi, “Chatbot and its Impact on the Retail Industry,” in *2025 3rd International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT)*, Bengaluru, India: IEEE (Conference Proceedings), 2025, pp. 2084–2089. doi: 10.1109/IDCIOT64235.2025.10915098.
- [21] A. D. Hartanto, Y. N. Kholik, and Y. Pristyanto, “Stock price time series data forecasting using the light gradient boosting machine (LightGBM) model,” *JOIV Int. J. Informatics Vis.*, vol. 7, no. 4, pp. 2270–2279, 2023, doi: dx.doi.org/10.62527/joiv.7.4.1740.
- [22] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, “M5 accuracy competition : Results , findings , and conclusions,” *Int. J. Forecast.*, vol. 38, no. 4, pp. 1346–1364, 2022, doi: 10.1016/j.ijforecast.2021.11.013.
- [23] M. Steve, S. Brightwood, and O. Godwin, “Implementing AI Chatbots for Real-Time Supply Chain Monitoring and Risk Management,” 2024. [Online]. Available: <https://www.researchgate.net/publication/383395365>
- [24] L.-C. Chen, H.-T. Weng, M. S. Pardeshi, C.-M. Chen, R.-K. Sheu, and K.-C. Pai, “Evaluation of prompt engineering on the performance of a large language model in document information extraction,” *Electronics*, vol. 14, no. 11, p. 2145, 2025, doi: 10.3390/electronics14112145.
- [25] H. J. Wahedi, M. Heltoft, G. J. Christophersen, T. Severinsen, S. Saha, and I. E. Nielsen, “Forecasting and inventory planning: An empirical investigation of classical and machine learning approaches for svanehøj’s future software consolidation,” *Appl. Sci.*, vol. 13, no. 15, p. 8581, 2023, doi: 10.3390/app13158581.