Integration of Predictive Analytics into Inventory Management Systems Using Machine Learning

José A. Martínez de La Cruz
Master of Engineering in Computer Engineering
Advisor: Dr. Nelliud Torres
Polytechnic University of Puerto Rico
Graduate Project EXPO – February 2026

Abstract – *Small and medium-sized retail businesses frequently face difficulties maintaining optimal inventory levels due to fluctuating demand, seasonality, and traditional manual decision-making. This article proposes the design of an intelligent inventory management system integrating predictive analytics through machine learning techniques to forecast product demand and generate dynamic restocking recommendations. The system utilizes a cloud-hosted PostgreSQL database managed through Supabase, a FastAPI-based Python backend, and a React-based web interface. Forecasting models are developed using established time-series methodologies such as ARIMA, Prophet, and Long Short-Term Memory (LSTM) neural networks. The goal is to demonstrate how predictive analytics can enhance operational efficiency, reduce losses caused by inadequate inventory planning, and support data-driven decision-making in modern retail environments.*

Keywords *– Cloud Databases, Inventory Management, Machine Learning, Predictive Analytics, Time-Series Forecasting.*

INTRODUCTION

Inventory management plays a critical role in retail operations by ensuring product availability, reducing operational costs, and preventing financial loss caused by stockouts or overstocking. Classical inventory theory emphasizes that inadequate replenishment strategies lead to increased holding costs, lost sales, and inefficiencies throughout the supply chain [1]. Traditional inventory systems commonly rely on fixed reorder points or manual estimations, which are limited in their ability to adapt to dynamic market conditions, seasonal variations, or sudden shifts in consumer behavior.

Recent technological advances have enabled businesses to integrate data-driven forecasting into everyday operations. Predictive analytics, powered by machine learning algorithms, can analyze historical sales data to identify trends, recognize seasonality, and estimate future product demand more accurately than conventional methods [2]. This shift from reactive to proactive inventory control allows managers to make informed decisions supported by quantitative evidence.

The adoption of cloud-based solutions has further enhanced the scalability, accessibility, and reliability of modern management systems. According to the National Institute of Standards and Technology (NIST), cloud computing provides "on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction" [3]. This architecture is ideal for retail environments requiring centralized data storage, real-time synchronization, and remote access across multiple locations.

In addition, this project considers basic evaluation metrics commonly used in forecasting. The Mean Absolute Error (MAE) measures how far the predictions are from the actual values on average, using simple differences between them. The Mean Absolute Percentage Error (MAPE) expresses this difference as a percentage, which helps interpret how large the prediction errors are relative to the real data. These measurements provide an easy way to compare different forecasting models and understand how accurate their predictions may be.

This article presents the design of InventNova, an intelligent inventory management system that integrates cloud computing and machine learning forecasting models. The system leverages a cloud-

hosted PostgreSQL database managed through Supabase, a FastAPI based backend for data processing, and a React web interface for visualization. Through time-series forecasting techniques such as ARIMA, Prophet, and LSTM neural networks, the platform aims to improve demand prediction accuracy, reduce inventory-related losses, and support digital transformation in small and medium-sized retail businesses [4].

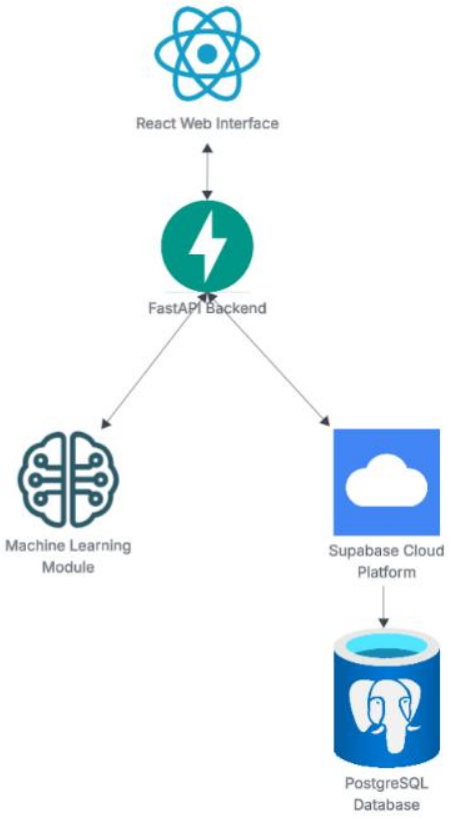

Figure 1
System Architecture Design

System Architecture

The proposed architecture consists of four main components:

1. **Data Layer:** A PostgreSQL database hosted on Supabase that stores raw transaction data and forecast results.
2. **Machine Learning Module:** A Python-based service responsible for data extraction, preprocessing, model training, and demand prediction.
3. **FastAPI Backend:** Facilitates communication between the database, machine learning module, and user interface; exposes REST endpoints such as demand forecasts and restocking recommendations.
4. **React Frontend:** Visualizes forecasts, product trends, and recommended reorder quantities in real time. This layered architecture ensures modularity, scalability, and maintainability.

Figure 1 illustrates the overall system architecture of the proposed inventory management platform, highlighting the interaction between the data layer, machine learning module, backend services, and web interface.

Theoretical Framework and Related Work

This section presents the theoretical foundations and related work that support the proposed system design. It reviews traditional inventory management approaches and introduces predictive analytics and machine learning techniques commonly applied to time-series forecasting in retail environments.

Inventory Management Systems

Inventory management systems track stock quantities, record transactions, and support replenishment decisions. Traditional approaches use static rules, such as triggering reorders when inventory falls below a predefined threshold. Although simple, these methods fail to incorporate real-time market dynamics or historical sales patterns, resulting in overstocking or stock outings.

Modern research highlights the importance of integrating data analytics into inventory processes. Studies demonstrate that machine learning techniques can significantly improve decision accuracy, especially when applied to time-series forecasting in retail environments.

Predictive Analytics and Machine Learning

Machine learning is a subfield of artificial intelligence concerned with the study of algorithms that improve their performance automatically

through experience, as defined by Mitchell [3]. These algorithms learn patterns from historical data and use them to make predictions or informed decisions without requiring explicit programming for each task. Predictive analytics apply these machine learning techniques to anticipate future outcomes.

- **ARIMA (Autoregressive Integrated Moving Average):** ARIMA is a classical statistical model used for analyzing and forecasting time-series data, which are observations recorded at regular time intervals where the order of the data is important because past values influence future values [4]. ARIMA captures linear relationships over time by combining three components: autoregression (relationship with past values), differencing (removing long-term trends), and moving averages (effects of past forecast errors).
- **Prophet:** Prophet is a forecasting framework developed by Meta that models seasonality, meaning patterns in the data that repeat at regular intervals such as daily, weekly, or yearly cycles [5]. These seasonal patterns often occur in retail, where sales increase during weekends, holidays, or specific times of the year. Prophet separates the time-series into trend, seasonality, and special events, making it effective for business data with recurring behaviors.
- **LSTM Neural Networks**: LSTM is a type of recurrent neural network designed to learn long-term dependencies in sequential data. A neural network is a computational model inspired by the human brain, made of interconnected processing units ("neurons") that learn by adjusting internal weights based on the data they observe [6]. LSTM networks improve upon traditional recurrent networks by preventing the loss of important information over long sequences, allowing them to model complex and nonlinear temporal patterns more effectively [7].

Figure 2 presents a conceptual framework summarizing the relationship between predictive analytics components and inventory management processes discussed in the related work.

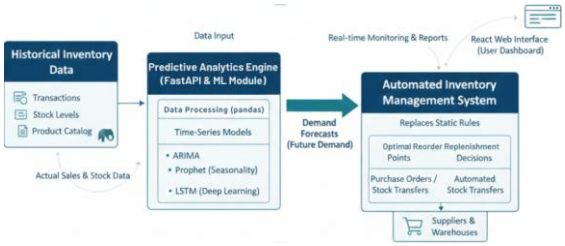

Figure 2
The Framework and Related Work

TECHNOLOGIES USED

Cloud Computing Technology

Cloud computing is a technological model that allows organizations to access computing resources such as storage, servers, networks, and applications through the internet without maintaining local hardware infrastructure. The National Institute of Standards and Technology (NIST) defines cloud computing as "on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction" [2]. This model provides scalability, reliability, remote availability, and efficient use of resources. In this article, cloud computing enables centralized storage of inventory data, real-time synchronization across multiple stores, and seamless integration with machine learning forecasting services.

PostgreSQL

PostgreSQL is an open-source relational database management system (RDBMS) that stores data using structured tables and supports the SQL query language. As an RDBMS, it organizes information into relations that can be queried, updated, and linked through primary and foreign keys. In the context of this project, PostgreSQL is used to store product records, inventory levels, transaction histories, and other operational data required by the system. These stored records constitute the historical information that serves as

input for generating demand forecasts through machine learning models [8].

Supabase

Supabase is a backend-as-a-service (BaaS) platform that provides a managed PostgreSQL database, user authentication services, file storage, and automatically generated application programming interfaces (APIs). It operates as a cloud-based service, meaning its resources are accessed remotely through the internet rather than hosted on local servers. In this project, Supabase is used to store application data, manage user access, and support communication between the system components through its API and database services [9].

FastAPI Backend

FastAPI is a Python framework used for creating application programming interfaces (APIs). A framework is a collection of tools and predefined structures that help developers build software by providing an organized way to handle common tasks, such as managing requests, processing data, and defining how different components interact. In this project, FastAPI functions as the application server that receives requests from the web interface, exchanges data with the database, and coordinates the execution of the machine learning prediction module [10] [11].

React Web Interface

React is a software library written in JavaScript. JavaScript is a programming language commonly used to create content that runs inside web browsers and allows web pages to display information, respond to user actions, and update content without reloading the entire page. React provides a structured way to build user interface components that display and update information.

In this article, React is used to create a web interface that presents inventory data, demand forecasts, and related information to the user. React organizes the visual elements of the system and manages the communication between the user interface and the application server.

Machine Learning Libraries

The predictive module relies on widely adopted Python libraries, which are collections of reusable code that provide ready-made functions and tools to simplify complex tasks. These libraries include:

- **Pandas** for data preprocessing and transformation.
- **Scikit-learn** for baseline modeling.
- **Stats models** for ARIMA implementations.
- **Prophet** for seasonal forecasts.
- **TensorFlow** for deep learning–based LSTM networks.

These tools enable comprehensive experimentation with multiple forecasting approaches. Together, these libraries create a flexible experimentation environment for comparing different forecasting strategies under a unified workflow.

METHODOLOGY

This section describes the methodological approach followed during the design of the proposed system. It outlines the main stages considered for data preparation, model selection, forecast integration, and visualization, providing a structured view of how predictive analytics would be incorporated into the inventory management workflow.

Data Preparation

In the proposed system, it is expected that historical sales data will be collected and organized by product and date. Once available, these data will undergo preprocessing steps that may include addressing missing values, identifying and handling outliers, encoding categorical information, and normalizing quantities when necessary. The objective of this stage is to prepare clean, consistent time-series datasets suitable for use in forecasting models.

Model Selection and Evaluation

Multiple forecasting methods are tested to determine which delivers the best accuracy. Evaluation metrics include Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE). These metrics allow objective comparison across models and products [12].

Forecast Integration

In the proposed design, the forecasting component is expected to generate predicted values that could later be stored in a dedicated table. This table may include fields such as product identifiers, forecast dates, estimated quantities, and other information needed for future analysis. As part of the system architecture, the backend is planned to include access points that would allow the user interface to request forecast.

Visualization and User Interaction

Within the planned interface design, the system is expected to present visual representations of forecast results to help users interpret potential inventory behavior. These visual elements may consist of charts, graphs, or indicators intended to highlight items that could require attention. The purpose of these visualizations is to offer a clear way for users to understand the predictive information once forecasts have been generated and integrated into the system. Figure 3 depicts the entity-relationship diagram (ERD) that represents how forecast data, products, and user interactions are structured within the proposed system architecture.

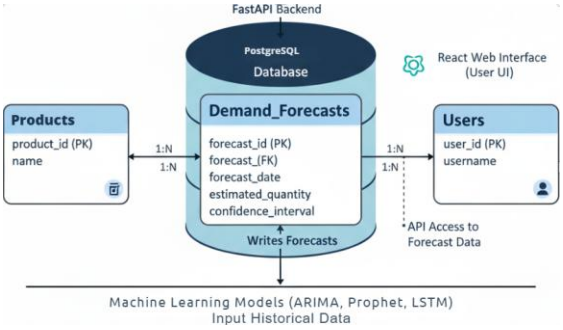

Figure 3
Forecast Integration Architecture Entity-Relationship Diagram (ERD)

EXPECTED RESULTS

The development of this article followed a structured design procedure for this project involved several integrated steps. First, system requirements were defined by identifying the essential components needed for an intelligent inventory management solution, including centralized data storage, support for multi-store environments, a forecasting module, and a user-facing web interface. These requirements guided the overall architecture, which was structured into four layers: a PostgreSQL database hosted on Supabase for cloud-based storage; a FastAPI backend responsible for API communication and data processing; a machine learning module developed in Python; and a React-based frontend for visualization and user interaction.

The second step consisted of reviewing existing forecasting methodologies applicable to retail inventory planning. This included classical statistical models such as ARIMA, decomposition-based frameworks like Prophet for handling seasonality, and deep learning architectures such as LSTM networks for capturing long-term temporal patterns. Each method was examined in terms of data assumptions, computational complexity, interpretability, and suitability for fluctuating retail demand.

The project then proceeded with the selection of appropriate tools for cloud integration, data processing, and predictive modeling. Supabase was chosen for reliable cloud-hosted PostgreSQL management and built-in authentication features; FastAPI was selected due to its scalability, asynchronous processing capabilities, and compatibility with machine learning workflows; and Python libraries such as pandas, scikit-learn, stats models, Prophet, and TensorFlow were incorporated to support preprocessing, exploratory modeling, and the conceptual configuration of forecasting techniques. The frontend design used React to outline how forecast outputs and inventory recommendations could be displayed in a modern web interface.

Although the project remains in the design stage, this procedure establishes a complete conceptual framework for integrating predictive analytics into an inventory management system. It clarifies how data would flow through the platform, how forecasting models could be embedded into operational workflows, and how cloud-based technologies would support scalability and future expansion.

CONCLUSION

This article presents the design of an intelligent inventory management system that incorporates cloud technologies and machine learning–based forecasting methods. The proposed architecture integrates data storage, predictive modeling, and user interaction into a unified structure intended to support data-driven decision-making in retail environments. While the system is presented at the design stage, its components outline a framework that may guide future implementation and evaluation.

Based on the proposed methodology, it is expected that the system could contribute to improving inventory planning by providing demand forecasts generated from historical data. As part of the design, conceptual screens and sample report layouts were created to illustrate how forecast results, inventory alerts, and replenishment indicators would be displayed to end users. These design elements demonstrate how predictive analytics could be incorporated into routine inventory workflows once historical data and development resources become available. Rather than presenting measured performance outcomes, the project defines the structural and visual components necessary for a system that could eventually support more informed inventory planning, multi-store scalability, and predictive decision support. Figure 4 shows a conceptual dashboard mock-up designed to illustrate how forecast results, inventory alerts, and performance indicators could be presented to end users.

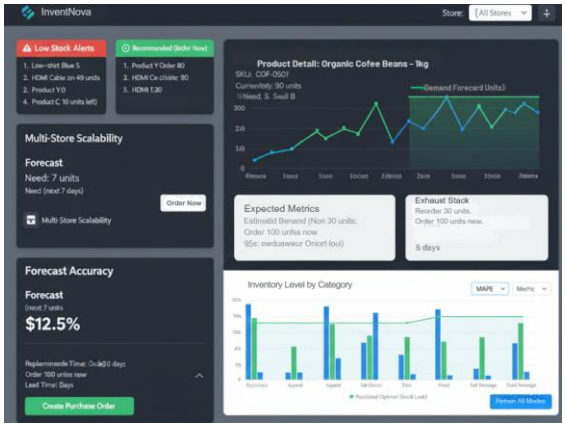

Figure 4
Conceptual Inventory Forecasting Dashboard (No Real data)

Future Work

Future work for this project includes several areas that extend beyond the scope of the current design phase but are essential for transforming the proposed architecture into a fully operational forecasting system. Once real transactional data becomes available, the forecasting models can be implemented, trained, and validated in a production environment to assess their performance under real-world inventory conditions.

Additional enhancements may include establishing automated model retraining pipelines that periodically update forecasting parameters as new sales data accumulates. This would allow the system to adapt to shifts in demand behavior without requiring manual adjustments. Future versions may also incorporate external variables such as seasonal patterns, promotional events, and supplier lead times, enabling the forecasting engine to generate more accurate and context-aware predictions.

Improvements to the user interface represent another important direction. The dashboard could be expanded to include advanced visual analytics, interactive charts, and customizable alerts that notify managers of potential stockouts, excess inventory, or unusual demand trends. These visualization tools would enhance the decision-support component of the system by making complex forecasting outputs easier to interpret.

Furthermore, deeper integration with supplier ordering workflows may be explored. This includes

features such as automated purchase order generation, recommended reorder quantities, expected delivery timelines, and synchronization with supplier systems. These capabilities would bridge the gap between forecasting and operational execution, ultimately moving the system toward a comprehensive, end-to-end inventory management solution.

REFERENCES

[1] E. Silver et al., *Inventory Management and Production Planning and Scheduling*, 3rd ed. Hoboken, NJ, USA: Wiley, 1998.

[2] P. Mell and T. Grance, *The NIST Definition of Cloud Computing*. Gaithersburg, MD, USA: Nat. Inst. Standards Technol., NIST Special Publication 800-145, 2011.

[3] T. Mitchell, *Machine Learning*, 1st ed. New York, NY, USA: McGraw-Hill, 1997.

[4] G. Box *et al.*, *Time Series Analysis: Forecasting and Control*, 5th ed. Hoboken, NJ, USA: Wiley, 2015.

[5] Meta Platforms Inc. (2023). *Prophet: Forecasting at Scale* [Online]. Available: https://facebook.github.io/prophet (accessed: Feb. 14, 2026).

[6] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.

[7] I. Goodfellow et al., *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[8] PostgreSQL Global Development Group. (2024). PostgreSQL Documentation [Online]. Available: https://postgresql.org/docs (accessed: Feb. 14, 2026).

[9] Supabase Inc. (2024). Supabase Documentation [Online]. Available: https://supabase.com/docs (accessed: Feb. 14, 2026).

[10] S. Tiago, *FastAPI: Modern Web Framework for APIs with Python, 1st ed*. Sebastopol, CA, USA: O'Reilly Media, 2022.

[11] M. Grinberg, *Modern Backend Development with Flask and FastAPI, 1st ed*. Birmingham, U.K.: Packt Publishing, 2021.

[12] R. Hyndman and G. Athanasopoulos. (2021). *Forecasting: Principles and Practice*, 3rd ed. [Online]. Available: https://otexts.com/fpp3 (accessed: Feb. 14, 2026).