


# FULL REPORT

# SMART INVENTORY MANAGEMENT AND SALES PREDICTION SYSTEM

Lucky Agarwal
Papimon Leelamali
Tanawat Jukmongkol

Department of Robotics and AI, School of Engineering
King Mongkut's Institute of Technology Ladkrabang
Academic Year 2025




# FULL REPORT

## SMART INVENTORY MANAGEMENT AND SALES PREDICTION SYSTEM

Lucky Agarwal
Papimon Leelamali
Tanawat Jukmongkol

Department of Nanoscience and Nanotechnology, College of Materials Innovation and Technology
Department of Robotics and AI, School of Engineering
King Mongkut's Institute of Technology Ladkrabang
Academic Year 2025

BY

1. Lucky Agarwal 67011698 Programming the Backend and UI / 33.33%
2. Papimon Leelamali 67011635 Prediction Model / 33.33%
3. Tanawat Jukmongkol 66011255 Database Management / 33.33%

PROMPT ENGINEERING SUBJECT FOR FINAL REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF BACHELOR OF ENGINEERING

ROBOTICS AND AI

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

ACADEMIC YEAR 2025

# ABSTRACT

This project presents the development of a smart inventory management and sales prediction system specifically designed for small and medium-sized enterprises (SMEs). The system aims to optimize stock levels, improve operational efficiency, and support data-driven decision-making. By integrating a robust backend, a structured database, and an interactive frontend, the system provides real-time inventory monitoring, automated alerts for low stock or overstock conditions, and predictive analytics for future demand forecasting. Using Meta's Prophet model, the system leverages historical sales data to generate accurate demand forecasts, allowing businesses to proactively manage procurement and sales strategies.

# ACKNOWLEDGEMENT

The successful completion of this research, titled ”Smart Inventory Management and Sales Prediction System,” is attributed to the support and collaboration of several key individuals and institutions.

We extend our profound gratitude to **King Mongkut’s Institute of Technology Ladkrabang** for providing the academic framework and resources necessary to undertake this project during the Academic Year 2025. We specifically thank the faculty and staff of the **Department of Robotics and AI, School of Engineering**, and the **Department of Nanoscience and Nanotechnology, College of Materials Innovation and Technology**, for their guidance and instruction.

We would also like to acknowledge the crucial contribution of our project team members:

- **Papimon Leelamali (67011635)**, who dedicated effort to the development of the Prediction Model.
- **Tanawat Jukmongkol (66011255)**, who managed the Database Management aspects of the system.
- **Lucky Agarwal (67011698)**, who focused on Programming the Backend and UI.

Each team member contributed 33.33% to the project’s execution.

Finally, we express our sincere appreciation to the small and medium-sized apparel businesses whose context and challenges drove this research, including **Toogtons Innovator Co., Ltd.** and **Pajara Company Limited**, whose operational data provided the necessary real-world foundation for developing the inventory management and sales prediction models.

# TABLE OF CONTENTS


Page

**Abstract** ..... i

**ACKNOWLEDGEMENT** ..... ii

**TABLE OF CONTENTS** ..... iv

**LIST OF TABLES** ..... v

**LIST OF FIGURES** ..... vi

**CHAPTER 1 INTRODUCTION** ..... **1**

1.1 Company Introduction ..... 1

1.1.1 Company Background ..... 1

1.1.2 Mission and Vision ..... 1

1.1.3 Nature of Business ..... 1

1.2 Problem State ..... 1

1.3 Research Objectives ..... 2

1.4 Scope of Research ..... 2

1.5 Methods of Conducting Research ..... 2

1.6 Expected Benefits ..... 2

**CHAPTER 2 CONCEPTS, THEORIES AND RELATED RESEARCH** ..... **3**

2.1 Concepts ..... 3

2.2 Related Theories and Research ..... 3

2.3 Model Framework ..... 4

2.3.1 Data Collection and Processing ..... 4

**CHAPTER 3 METHODOLOGY** ..... **5**

3.1 Design Method ..... 5

3.1.1 System Architecture ..... 5

3.1.2 Development Stack ..... 6

3.2 Conceptual Stage ..... 7

3.2.1 Prophet Model Overview ..... 7

3.2.2 Trend Function ..... 7

3.2.3 Seasonality Function ..... 7

3.2.4 Holiday Function ..... 8

3.2.5 Error Term ..... 8

3.2.6 Model Evaluation ..... 8

3.3 Implementation Stage ..... 9
3.3.1 Inventory Management Page ..... 9
3.3.2 Inventory Summary Page ..... 10
3.3.3 Product Detail Page ..... 14
3.3.4 Sales Prediction Page ..... 15
3.4 Control Integration Stage ..... 16
**CHAPTER 4 RESULTS ..... 17**
4.1 Project Results ..... 17
4.1.1 Forecasting Performance ..... 17
4.1.2 System Implementation and Outcomes ..... 17
4.1.3 User Feedback and Validation ..... 17
**CHAPTER 5 DISCUSSION, CONCLUSION, AND SUGGESTIONS ..... 19**
5.1 Discussion ..... 19
5.2 Conclusion ..... 19
5.3 Suggestion ..... 19

# LIST OF TABLES

**Table** **Page**

3.1 Inventory Health Scoring Criteria ..... 11

# LIST OF FIGURES

**Figure** **Page**


3.1 System Architecture Diagram ..... 5
3.2 Database Schema ..... 6
3.3 Inventory Management Page ..... 9
3.4 Settings Panel ..... 10
3.5 Inventory Summary Page ..... 11
3.6 Category Sales Volume ..... 12
3.7 Individual Category Sales ..... 12
3.8 Top Performing SKUs ..... 13
3.9 Overall Sales Analytics ..... 13
3.10 Overall Sales by Channel ..... 14
3.11 Product Detail Overview ..... 14
3.12 Channel Sales Breakdown ..... 15
3.13 Sales Prediction Page ..... 16

# CHAPTER 1

# INTRODUCTION

## 1.1 Company Introduction

### 1.1.1 Company Background

This project is done in collaboration with two companies, Toogton and Pajara. Toogtons Innovator Co., Ltd. is a Thai apparel company founded in 2023 in Bangkok by Nirut Sae-Eia. The brand Toogtons (ทุกตอนส์) started from street sales and is now known for its ultra-soft "marshmallow" boxers and casualwear. Meanwhile, Pajara Company Limited is a Thai apparel company registered on 2 April 2014 and based in Bangkok.

### 1.1.2 Mission and Vision

Toogtons aims to redefine every comfort with not only accessible but also soft and stylish clothing. The company aims to be Southeast Asia's leading comfortwear brand, known for prioritizing comfort, quality and customer-focused designs. Pajara aims to appeal to a broad audience with a plethora of recognizable brand name characters on both comfortable casual wear and sleepwear. The company strives to be a pioneer in producing and distributing officially licensed apparel with a focus on quality and comfort.

### 1.1.3 Nature of Business

Toogton engages in the design, production and online retail of undergarments and casual wear, primarily distributed through online stores such as Shoppee, Lazada and their official website as well as physical stores across Thailand. Pajara similarly engaged in the design, manufacturing and the wholesale of apparel for various demographics with an emphasis on producing outerwear featuring licensed characters such as Mickey Mouse or Stitch under the brand "Pajara Pajamas".

## 1.2 Problem State

1. Both companies lack access to any system that would allow them to perform real-time stock tracking. Without insight to stock movement, the company risks experiencing stockouts and missing out on sales that come through.
2. Because the companies lack insights on certain factors of their sales and stocks, reordering stock is not optimized. In order to maximise profits and stock uptime, a more robust and efficient reordering system must be implemented.
3. Both companies lack sales forecasting which robs them of useful insights into the possible future of their sales. Having additional insights into future sales may positively influence upcoming sales decisions.
4. There is no centralized system where all SKUs can be displayed with their statuses visible. As a result, underperforming SKUs may not be detected and properly dealt with.
5. Manually compiling all the sales and stocks data is time consuming, expensive and prone to error. An automated system to sort out all the data would cut the operational costs considerably.

## 1.3 Research Objectives

1. An interactive inventory dashboard will be implemented to display stock levels and products. Users will be able to dynamically interact with and customize the dashboard according to their preferences.
2. The application will also feature a predictive sales forecasting system using historical sales data provided by the company. Users will be able to specify the prediction duration, as well as select the desired SKU, category, or sales channel.
3. An automated low-stock alert system will be incorporated into the application. Users will receive notifications to restock whenever the stock level of a product falls below a manually set threshold.
4. SKUs will be cleaned and dynamically categorized for a more comprehensible viewing experience. Products will automatically be sorted into active stock and dead stock categories.

## 1.4 Scope of Research

1. The interactive inventory dashboard will be developed utilizing a FastAPI backend and Next.js frontend to enable real-time data interaction and seamless user experience.
2. The open-source Prophet model will be utilized for sales forecasting, accompanied by a visual display for prediction accuracy and performance metrics.
3. Low-stock SKUs will not only be displayed in the notification alert system but will also be shown on the Restock Management Page of the application for easier inventory control.
4. Raw data obtained from the company will undergo an ETL (Extract, Transform, Load) process and be systematically grouped into categorized datasets for improved analysis and visualization.

## 1.5 Methods of Conducting Research

Reactjs will be utilized for the frontend of the application while fastAPI will be used for the backend with Supabase acting as the database for the application. Electron will also be utilized to render the desktop app. Prophet will be implemented for the sales forecasting system.

## 1.6 Expected Benefits

1. The cleaned data will be displayed on the dashboard, providing users with a visually pleasing representation of sales data and allowing them to gain valuable insights more easily.
2. With an automated low-stock alert, users will be able to efficiently and optimally reorder stock based on any stock threshold they desire.
3. Utilizing the sales forecasting system, users will be able to make well-informed business decisions using the insights provided by the predictive analysis.

# CHAPTER 2

# CONCEPTS, THEORIES AND RELATED RESEARCH

## 2.1 Concepts

In modern data-driven business environments, predictive modeling plays a crucial role in supporting strategic decision-making. One of the most common applications is in inventory management and sales forecasting, where accurate demand prediction enables organizations to maintain optimal stock levels and minimize operational costs.

The fundamental concept behind this work is to integrate a *data pipeline* with a *predictive model* to create a continuous flow of information—from raw data acquisition to actionable insights. This approach supports data integrity, consistency, and reliability while allowing machine learning algorithms to generate forecasts based on historical trends and seasonal variations.

The goal of our system is to enhance business efficiency through automation and predictive analytics. It aims to identify patterns in past sales data, anticipate future demand, and support better inventory and procurement planning. These concepts form the foundation of the **Model Framework** discussed in Section 2.3.

## 2.2 Related Theories and Research

Time-series forecasting is a well-established research domain that has evolved with the advancement of statistical methods and machine learning techniques. Traditional models such as the *Autoregressive Integrated Moving Average* (ARIMA) model and its variants (e.g., SARIMA) have been widely used for trend and seasonality detection. However, these models often require extensive manual parameter tuning and assume linear relationships among variables.

Recent developments in machine learning have introduced more flexible frameworks for time-series prediction. Among these, the **Prophet model**, developed by Meta (formerly Facebook), has gained attention due to its interpretability, scalability, and robustness to missing or irregular data. Prophet is based on an additive model that decomposes a time series into trend, seasonal, and holiday components, represented as:

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t \tag{2.1}$$

where $g(t)$ denotes the trend component, $s(t)$ captures seasonal variations, $h(t)$ accounts for special events or holidays, and $\epsilon_t$ is the random error term. This approach enables analysts to model complex patterns without deep statistical expertise.

In addition to Prophet, related studies have explored the use of neural networks (e.g., LSTM and GRU architectures) for sales forecasting. While deep learning models can provide higher accuracy with large datasets, Prophet offers superior transparency and faster implementation for small to medium-sized businesses.

## 2.3 Model Framework

The prediction framework in this study comprises two principal components: **the data pipeline** and **the predictive model**. Together, these components form an integrated system for generating accurate sales forecasts and supporting data-driven inventory management.

### 2.3.1 Data Collection and Processing

The data component follows a standardized *data engineering pipeline*, which ensures the quality, consistency, and reliability of the input used for forecasting. The process involves several stages:

1. **Data Collection:** Sales transaction data, stock data, and related metadata are uploaded as CSV or Excel files.
2. **Extraction and Transformation:** The raw data is extracted from source systems and transformed into a consistent schema. This stage often includes handling missing values, filtering out anomalies, and converting temporal data into uniform time intervals.
3. **Loading and Normalization:** Cleaned and structured data is loaded into a centralized database. Data normalization techniques are applied to remove redundancy and maintain referential integrity across different entities.
4. **Data Analysis:** Preliminary statistical analysis is performed to identify patterns, seasonality, and outliers in the dataset. These insights inform feature selection and model tuning.
5. **Data Prediction:** The processed data is fed into the predictive model for training and forecasting.

# CHAPTER 3

# METHODOLOGY

## 3.1 Design Method

### 3.1.1 System Architecture

The system architecture is designed to facilitate automated data processing, sales prediction, and analytical visualization for small and medium-sized businesses. The overall workflow consists of multiple integrated components that function cohesively to ensure seamless data flow and reliable forecasting results.

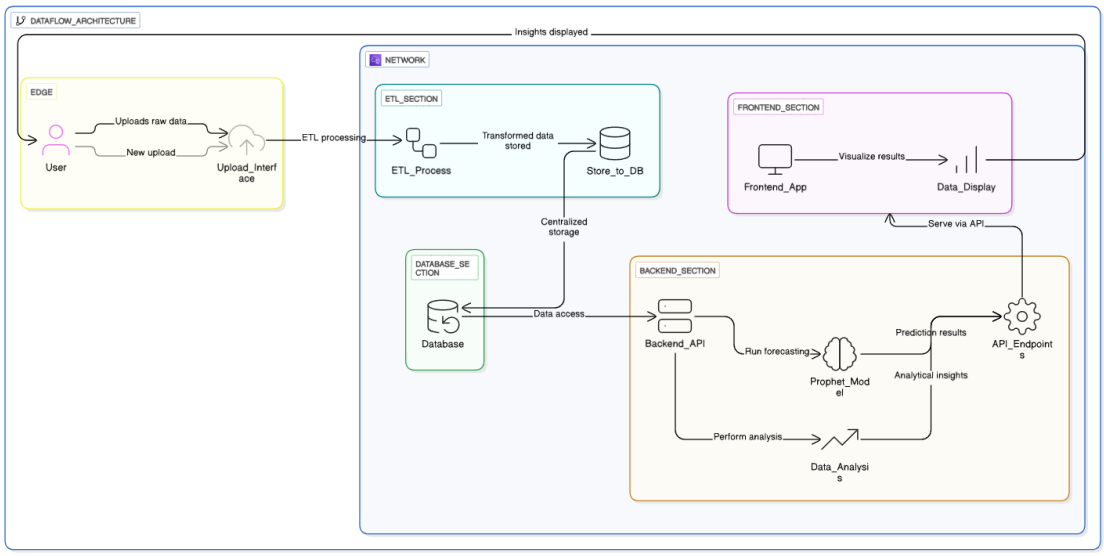



Figure 3.1: System Architecture Diagram

Raw sales data are initially uploaded in the form of Comma-Separated Values (CSV) files. These files undergo an Extract, Transform, and Load (ETL) process, which includes data cleaning, normalization, and validation to ensure consistency and quality. The cleaned and structured data are then stored in a centralized database for further processing. The database stores data in 3 tables: sales, stock and SKUs. The database schema is shwon below:

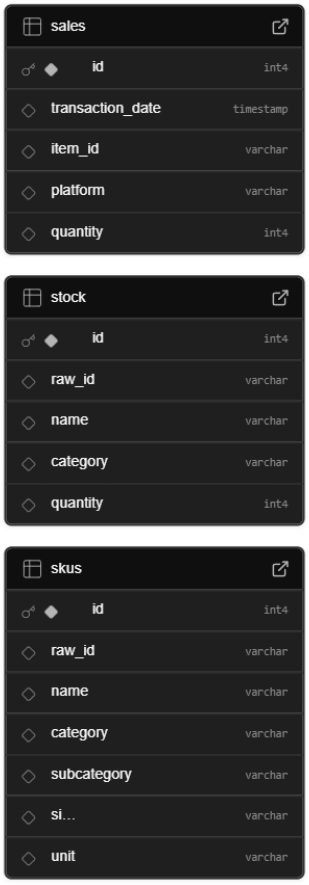



Figure 3.2: Database Schema

The backend component is responsible for retrieving data from the database, performing sales prediction, and executing analytical computations. The system analyzes the data generating key insights and demand predictions based on historical sales trends and seasonal patterns. These results are subsequently provided to the frontend interface for user interaction and visualization.

The frontend, implemented as a desktop application, presents users with an interactive dashboard to monitor inventory levels, analyze sales performance, and review predictive insights. The interface allows real-time visualization of key metrics and sales forecasts, enhancing decision-making efficiency.

### 3.1.2 Development Stack

The system uses a 3-tier architecture. The backend is developed using the FastAPI framework, chosen for its high performance and ease of integration with machine learning models. Data storage and management are handled through Supabase, an open-source backend-as-a-service platform providing a robust PostgreSQL database and authentication services.

For the frontend, React is utilized for its component-based architecture and responsiveness, while Electron enables the deployment of the system as a cross-platform desktop application. Together, these technologies provide a consistent and user-friendly experience across operating systems.

The sales prediction engine is implemented using the Prophet model, which is known for its capability to handle time series data with strong seasonal effects and missing observations. The integration of

Prophet within the FastAPI backend ensures that forecasting results can be efficiently computed and served to the user interface in near real time.

## 3.2 Conceptual Stage

The conceptual stage of this project focused on the theoretical foundation and model selection process for predictive sales forecasting. Extensive research was conducted on various time-series forecasting techniques, including statistical and machine learning approaches. After careful evaluation, the **Prophet** model, developed by Meta, was chosen due to its robustness in handling non-linear trends, seasonality, and missing data.

### 3.2.1 Prophet Model Overview

The Prophet model decomposes time-series data into four main components: trend, seasonality, holidays, and error. The general form of the model is expressed as:

$$y(t) = g(t) + s(t) + h(t) + \varepsilon_t \tag{3.1}$$

where:

- $y(t)$ represents the forecasted value at time $t$,
- $g(t)$ represents the **trend function**, describing non-periodic changes,
- $s(t)$ represents the **seasonality function**, capturing periodic fluctuations,
- $h(t)$ represents the **holiday effect**, and
- $\varepsilon_t$ is the **error term**, accounting for noise or randomness.

### 3.2.2 Trend Function

Prophet models the trend component using a piecewise linear or logistic growth function, allowing for flexibility in representing changes in direction or growth rate. Changepoints are introduced to identify where the trend significantly shifts. The trend function for a piecewise linear model can be represented as:

$$g(t) = \begin{cases} \beta_0 + \beta_1 t, & \text{if } t \leq c \\ \beta_0 - \beta_2 c + (\beta_1 + \beta_2)t, & \text{if } t > c \end{cases} \tag{3.2}$$

where $c$ represents the changepoint, $\beta_0$ is the intercept, and $\beta_1$, $\beta_2$ are the slopes before and after the changepoint, respectively.

### 3.2.3 Seasonality Function

Seasonal variations in sales are modeled using a Fourier series expansion, which captures cyclic patterns such as weekly, monthly, or yearly seasonality. The seasonality component is defined as:

$$s(t) = \sum_{n=1}^{N} \left( a_n \cos\left(\frac{2\pi n t}{P}\right) + b_n \sin\left(\frac{2\pi n t}{P}\right) \right) \tag{3.3}$$

where:

- $P$ is the periodicity of the season (e.g., 7 for weekly, 365.25 for yearly),

- $N$ is the order of the Fourier series, which determines the smoothness of the curve,
- $a_n$ and $b_n$ are coefficients learned from the data.

A lower order ($N$) provides a smoother approximation, while a higher order allows for capturing more complex seasonal patterns.

### 3.2.4 Holiday Function

The holiday component adjusts forecasts based on the historical effects of special events or holidays that impact sales. Prophet introduces binary indicator variables corresponding to each holiday date, which modify the prediction through additive effects. This is formulated as:

$$h(t) = \sum_{i=1}^{L} \kappa_i D_i(t) \tag{3.4}$$

where $L$ is the number of holidays, $\kappa_i$ is the learned impact of holiday $i$, and $D_i(t)$ is an indicator function that equals 1 if date $t$ corresponds to holiday $i$, and 0 otherwise.

### 3.2.5 Error Term

The residual component $\varepsilon_t$ represents stochastic noise or unexplained variability in the data. It is assumed to follow a normal distribution with zero mean:

$$\varepsilon_t \sim \mathcal{N}(0, \sigma^2) \tag{3.5}$$

This accounts for randomness not captured by trend, seasonality, or holiday effects.

### 3.2.6 Model Evaluation

To assess forecasting accuracy, the dataset is split into training and testing subsets. The model is trained on the historical (training) data and validated on the testing data. The performance is evaluated using statistical error metrics such as Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE), defined as follows:

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |A_i - F_i| \tag{3.6}$$

$$\text{MAPE} = \frac{100}{n} \sum_{i=1}^{n} \left| \frac{A_i - F_i}{A_i} \right| \tag{3.7}$$

where $A_i$ denotes the actual observed value, $F_i$ the forecasted value, and $n$ the total number of data points.

The combination of these mathematical components enables the Prophet model to effectively decompose, interpret, and predict future sales patterns with high interpretability and reliability.

## 3.3 Implementation Stage

During the implementation stage, we used the research done in the earlier phases to guide our development process. Each team member was assigned specific roles: Tanawat focused on backend and API development, Punpun led the UX/UI design and AI prediction framework, while Lucky was responsible for database management, frontend development, and the implementation of the stock notification system. We worked in parallel and merged our works to ensure fast development.

We designed our application with multiple pages showing different analysis and data visualiazation to enhance user experience and provide comprehensive insights into inventory and sales performance.

### 3.3.1 Inventory Management Page

First, we have the inventory management page where the users can upload the data and see the list of their products.

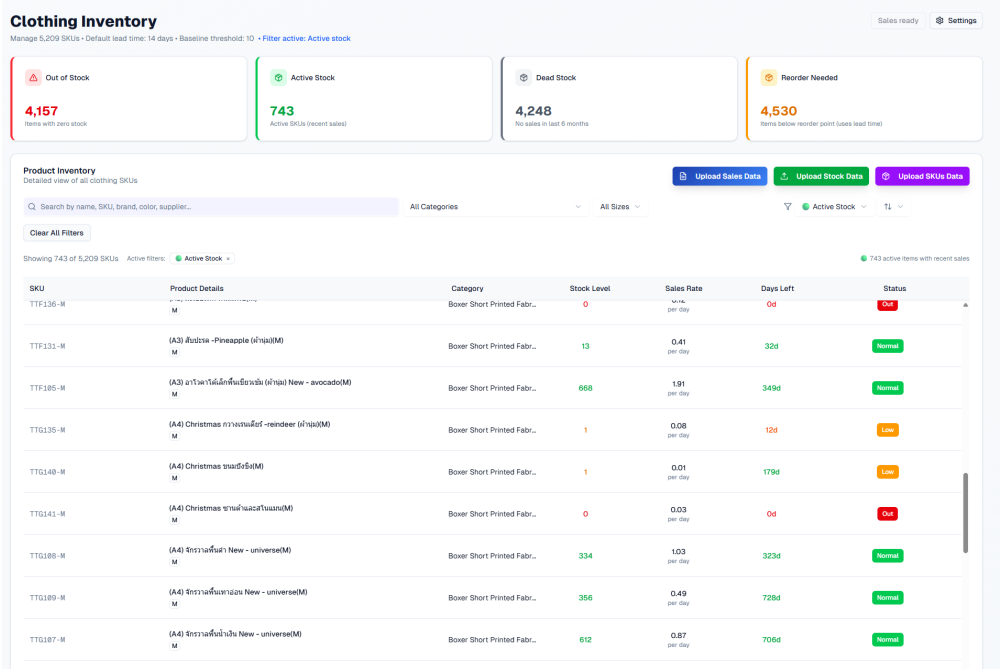



Figure 3.3: Inventory Management Page

4 quick summary cards are shown at the top showing the Out of stock, Active/Dead stock and Reorder needed count. Users can search for any specific product by searching by either product name or SKU ID. In addition, users can filter based on categories, sizes and stock levels. Each product entry displays key information such as stock level, sales rate, days left and status indicators to quickly assess inventory health. Key values are determined as follows:

- **Out of Stock**: Products with zero stock levels.
- **Active/Dead Stock**: Products are classified as active if they have had sales within the sales rate window; otherwise, they are considered dead stock.
- **Reorder Needed**: Products where current stock levels fall below the predefined reorder point. The reorder point is calculated as Reorder Point = Threshold + (LeadTime * DailySalesRate).

- **Sales Rate**: Total Sales / Period of sales.
- **Days Left**: Stock / Sales Rate.
- **Status Indicators**:
  - **Critical (Red)**: Stock = 0.
  - **Urgent (Red)**: Days left < 7.
  - **Low (Orange)**: Stock < Baseline Threshold.
  - **Watch (Yellow)**: Days Left < 30 days.
  - **Normal (Green)**: Otherwise.

The Baseline Threshold, Lead Time, Active/Dead stock window and Sales Rate window can be configured by the user in the settings page according to their business needs.

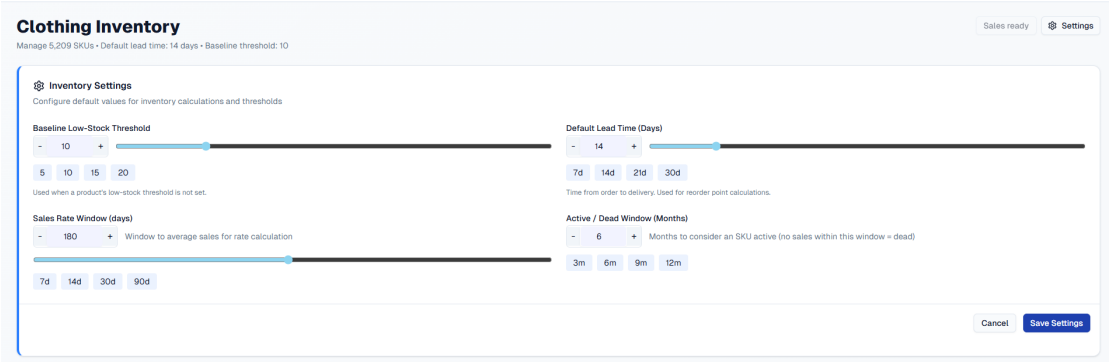



Figure 3.4: Settings Panel

### 3.3.2 Inventory Summary Page

Next we have the inventory summary page which provides an overview of the inventory health and performance metrics.

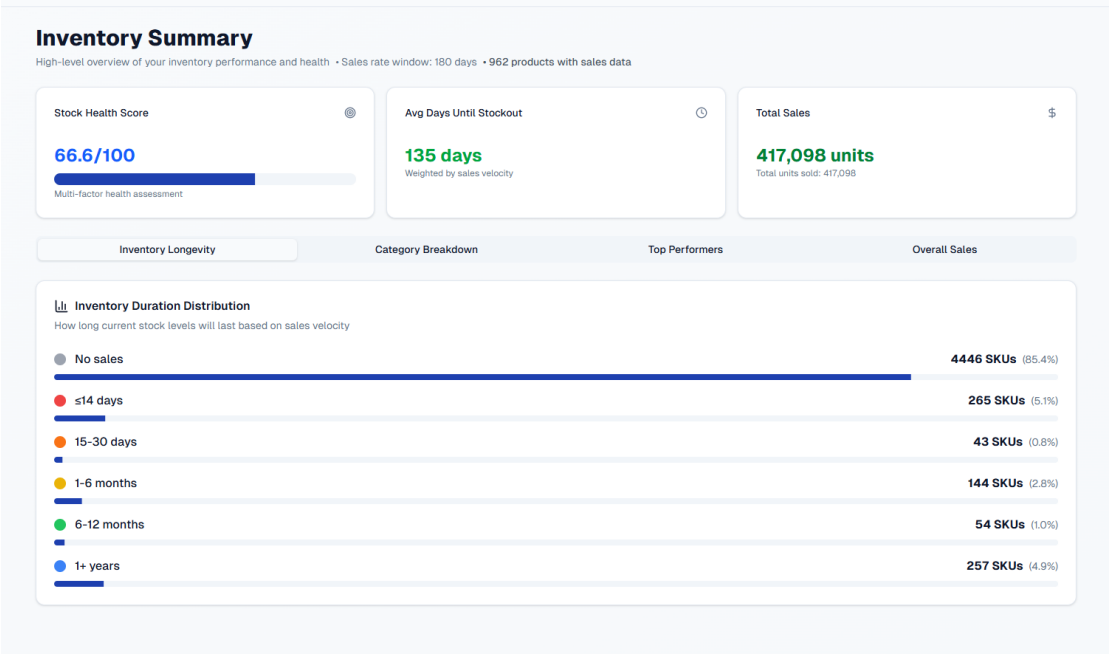



Figure 3.5: Inventory Summary Page

Here the users can see the overall stock health score calculated based on stock coverage, stock vs reorder point, sales activity and overstock status. A score for each SKU is determined and an average is taken to give an overall inventory health score out of 100. The criteria is shown below:

Table 3.1: Inventory Health Scoring Criteria

| Factor | Max Points | Criteria |
|---|---|---|
| Stock Coverage | 40 | Days until stockout: 90+ days = **40pts**, 60-89 = **30pts**, 30-59 = **20pts**, 14-29 = **10pts**, <14 = **0pts** |
| Stock Vs Reorder Point | 30 | Stock $\geq$ 2x reorder = **30pts**, $\geq$ 1.5x = **20pts**, $\geq$ 1x = **10pts**, $\leq$ reorder = **0pts** |
| Sales Activity | 20 | Last sale $\leq$ 7 days = **20pts**, $\leq$ 30 = **15pts**, $\leq$ 60 = **10pts**, $\leq$ 90 = **5pts**, $\geq$ 90 = **0pts** |
| Not Overstocked | 10 | Days left $\leq$ 365 = **10pts**, $\leq$ 500 = **5pts**, $\geq$ 500 = **0pts** |
| **Total** | **100** | Average across all products |

Average days until stockout is also shown which is a weighted average of the days left for each product weighted by their sales rate. In addition, the page has 4 tabs showing the inventory longevity, category breakdown, top performers and overall sales analytics. In the **inventory longevity** tab, skus are grouped based on their days left into different buckets to help users identify products that need attention. The criteria is as follows:

- **No sales:** dailySalesRate ≤ 0 or currentStock = 0
- **≤14 days:** daysUntilStockout ≤ 14
- **15–30 days:** daysUntilStockout > 14 and ≤ 30
- **1–6 months:** daysUntilStockout > 30 and ≤ 180
- **6–12 months:** daysUntilStockout > 180 and ≤ 365
- **1+ years:** daysUntilStockout > 365

In the **category breakdown** tab, graphs are shown for the sales volume by category and the individual category sales where the user can choose the category. These graphs can be set to weekly or monthly views. A list of the top 5 categories by sales volume is also shown.

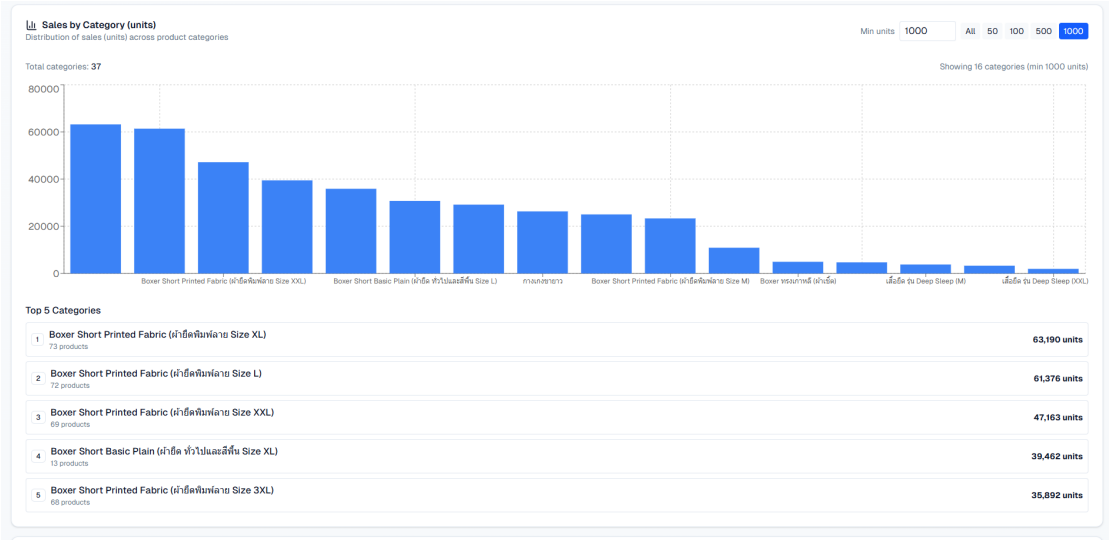



Figure 3.6: Category Sales Volume

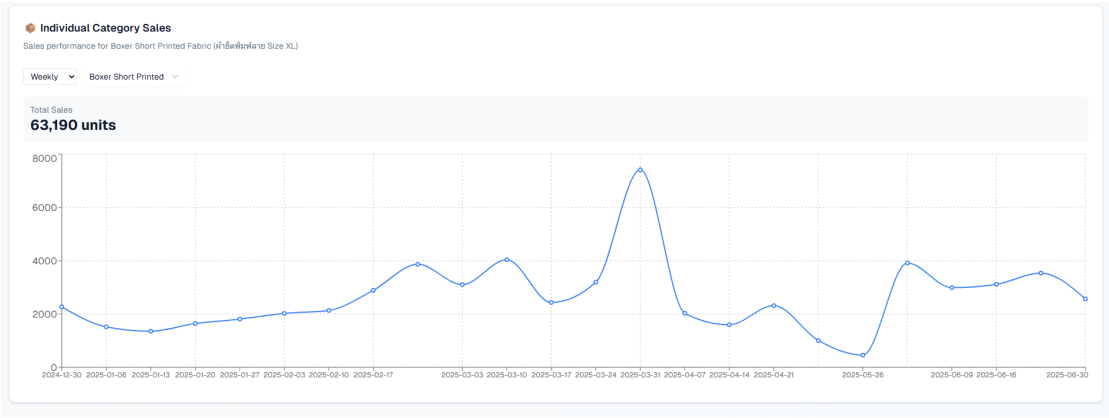



Figure 3.7: Individual Category Sales

A list of the top best selling SKUs is shown in the **top performers** tab and the user can choose the number of SKUs to display.

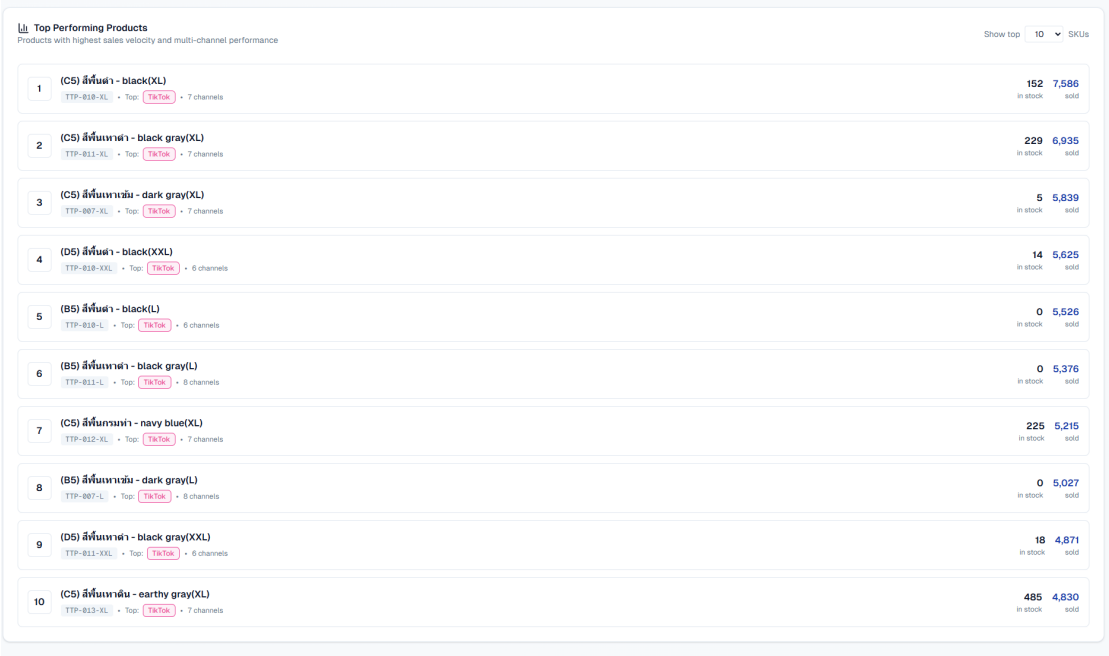



Figure 3.8: Top Performing SKUs

Finally, the **overall sales** tab shows the total sales over time, the average monthly sales and the month on month growth rate. It also shows the sales by channel where the user can choose all channels or individual channels.

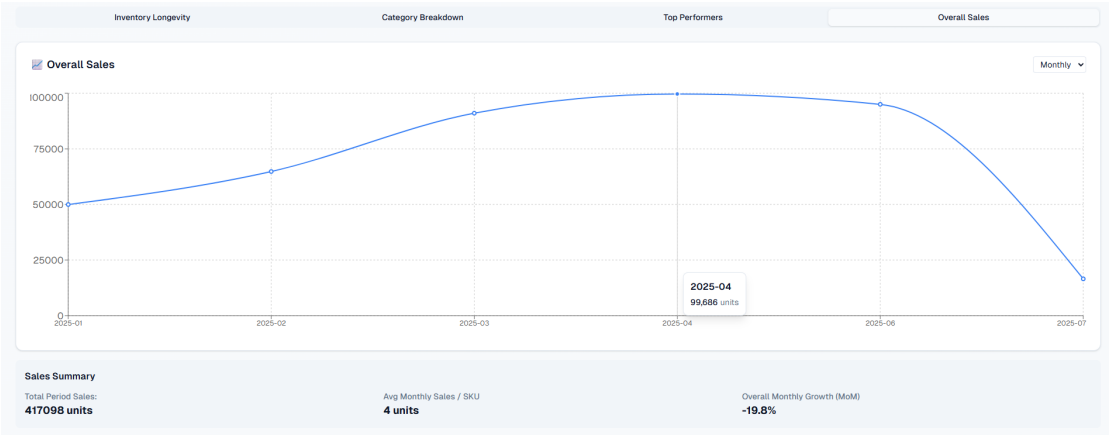



Figure 3.9: Overall Sales Analytics

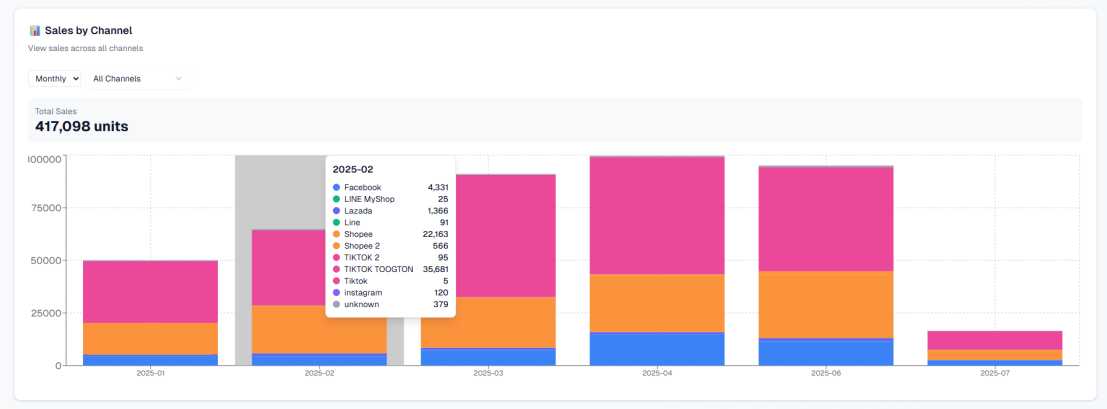



Figure 3.10: Overall Sales by Channel

### 3.3.3 Product Detail Page

The product detail page provides an in-depth view of individual SKU performance, inventory status, and KPIs.

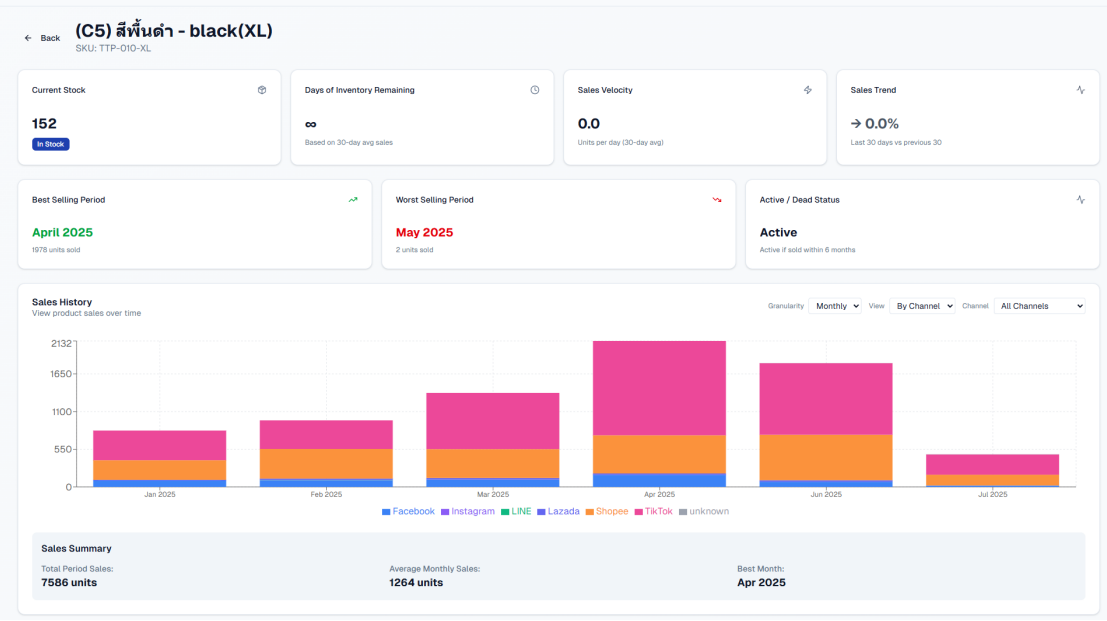



Figure 3.11: Product Detail Overview

Key performance indicators such as current stock, days of inventory remaining based on sales rate of the last 30 days, sales rate, sales trend based on the past 30 days vs the previous 30 days, best and worst selling period and the active/dead stock status are prominently displayed at the top. Then, a graph of the sales history over time is shown to visualize sales trends. This graph can be configured to show overall sales or sales by channel and can be set to weekly or monthly views.

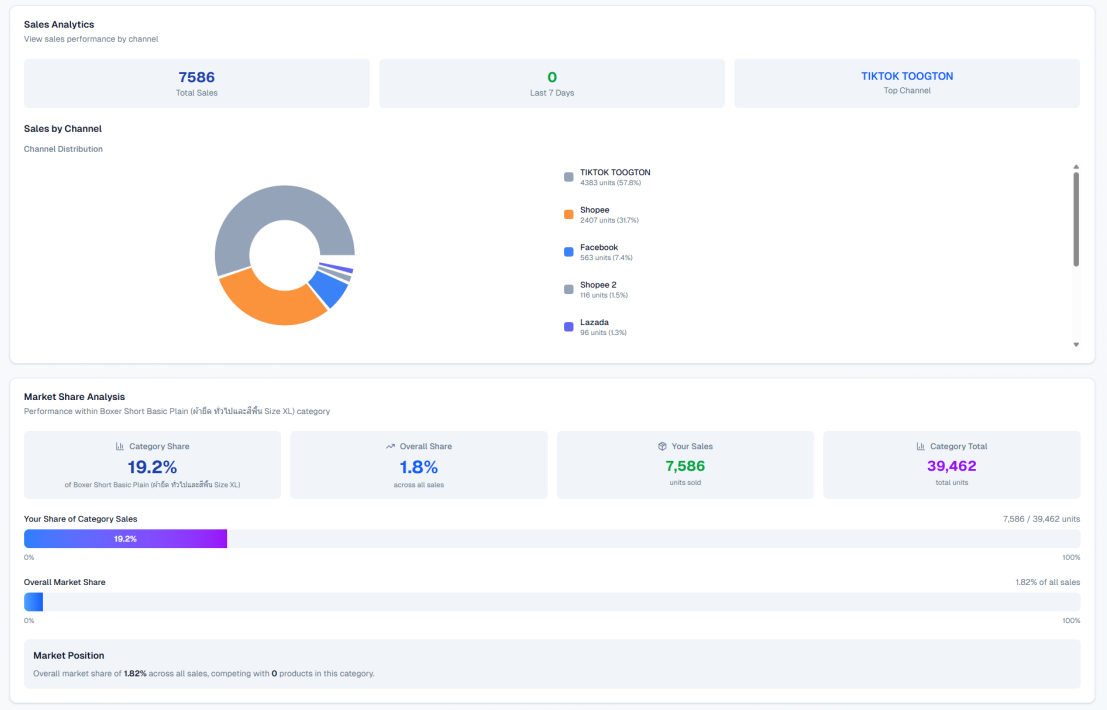



Figure 3.12: Channel Sales Breakdown

Below the sales history graph, a breakdown of sales by channel is presented, allowing users to analyze performance across different sales avenues. A market share analysis is also shown, illustrating the contribution of the SKU in its category and overall sales.

### 3.3.4 Sales Prediction Page

The sales prediction page provides users with forecasted sales data based on historical trends using the Prophet model. The users can choose predicions based on SKU, channel or category and the period for which they want the predictions.

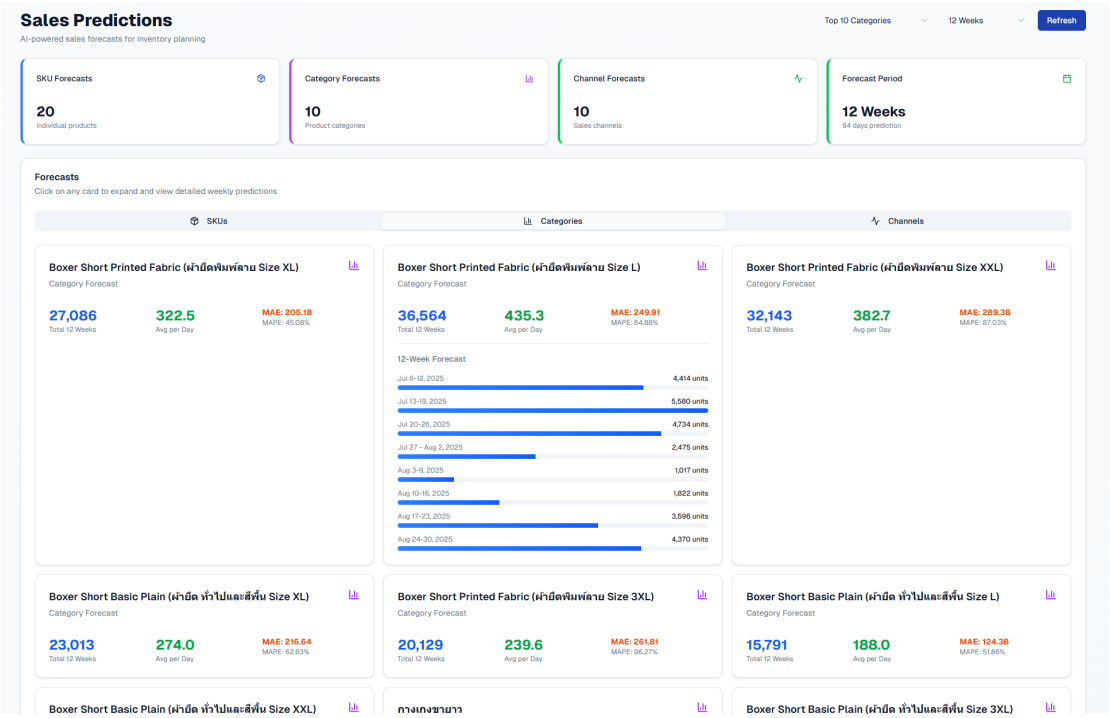



Figure 3.13: Sales Prediction Page

## 3.4 Control Integration Stage

The control and integration stage concentrated on ensuring system consistency, version management, and collaborative code maintenance. The team utilized Git as the primary version control system to manage source code, track changes, and coordinate development efforts among members. Through Git repositories and branching strategies, updates from different contributors were efficiently merged, minimizing conflicts and maintaining project stability.

This approach enabled continuous integration and testing throughout the development cycle, ensuring that new features and modifications could be safely incorporated into the system. The adoption of Git also facilitated transparent collaboration and traceability of development activities, contributing to the overall reliability and maintainability of the project.

# CHAPTER 4

# RESULTS

## 4.1 Project Results

The results of the project demonstrate the successful implementation of an integrated inventory management and sales prediction system. The developed application provides accurate forecasting, real-time stock tracking, and automated insights that enhance decision-making for small and medium-sized businesses. The evaluation of results focuses on both predictive performance and system functionality.

### 4.1.1 Forecasting Performance

The system's forecasting engine, based on the Prophet model, was trained on historical sales data that had been preprocessed through an ETL pipeline to remove inconsistencies and normalize values. The model effectively captured overall sales trends and seasonal demand fluctuations, providing reliable short- and medium-term predictions.

Quantitative evaluation using Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE) indicated low error values across product categories, confirming the model's suitability for inventory prediction tasks. Minor deviations between predicted and actual sales were attributed to data sparsity and market fluctuations.

### 4.1.2 System Implementation and Outcomes

Beyond forecasting, the implemented system achieved all targeted operational goals and demonstrated stable performance. Users can upload raw sales data in CSV format, which is automatically processed through the ETL pipeline. The backend (FastAPI and Supabase) handles forecasting requests and manages data storage, while the frontend (React and Electron) provides an interactive dashboard for monitoring sales, inventory, and forecast insights.

The final desktop application successfully performed:

1. Accurate forecasting of sales and stock demand.
2. Automated data cleaning and transformation during uploads.
3. Generation of real-time alerts for low or critical stock levels.
4. Visualization of analytics through an intuitive, responsive dashboard.

Empirical testing confirmed that the system improved visibility into stock movement and assisted in more effective inventory planning.

### 4.1.3 User Feedback and Validation

Both participating companies expressed positive feedback regarding the system's usability and practical value. During testing and demonstration sessions, users noted that the application provided clear, actionable insights into stock performance and sales trends. They appreciated the simplicity of uploading raw data and receiving automated analytics without requiring technical expertise.

Overall, the feedback confirmed that the developed solution met real-world operational needs

and demonstrated potential for wider adoption among small and medium-sized enterprises seeking affordable, data-driven inventory management tools.

# CHAPTER 5

# DISCUSSION, CONCLUSION, AND SUGGESTIONS

## 5.1 Discussion

The developed stock management and prediction system effectively met the project objectives by integrating automated data handling, analysis, and forecasting in a single platform. During testing, the system accurately processed and cleaned Excel and CSV files, removing inconsistencies such as blank fields and duplicated entries. The database integration ensured that large volumes of sales and stock data were stored efficiently and retrieved quickly for analysis.

The analytics dashboard allowed users to easily monitor inventory performance and sales trends, clearly identifying low-stock and out-of-stock items. Moreover, the sales prediction model provided meaningful forecasts that reflected general seasonal demand patterns. However, the accuracy of the prediction model varied and in some cases it had a high mean average error.

While the system performed well overall, it still relies on the user constantly uploading data manually into the system to keep it upto date. This can be a tedious thing to do recurrently and better ways to integrate data can be explored. Moreover, the ETL system to clean the data isn't fully robust and only works with specific pre-defined data formats.

Packaging the app was also a challenge as we encountered a lot of problems along the way with the app working on some devices and not working in other devices. Despite these challenges and limitations, we managed to meet all the requirements and deliver a complete solution.

## 5.2 Conclusion

In conclusion, we successfully developed and packaged a fully functional stock management app that allows users to input data in excel or CSV format,which is then cleaned, validated, and stored in a secure database.users can monitor stock and sales analytics, identifying out-of-stock and low-stock SKUs, as well as analyzing individual or overall inventory performance. Furthermore, the system integrates predictive analytics that generates future sales forecasts for specific time periods, categorized by SKU, product category, or sales channel.

## 5.3 Suggestion

To further enhance the functionality of the system, the following things can be implemented:

- **Integration with Internal Systems**: By connecting our system directly with the company's internal system which handles their stock and sales, it would allow our app to provide real time insights and monitoring without requiring the user to manually upload files every time.
- **Fine tuning prediction model**: By further fine tuning our prediction model, we can provide significantly better sales prediction which would be more beneficial to the company.
- **More robust ETL system**: A more robust ETL system would better clean the data and could work with any company's data once the format of the data is provided in the system.

- **Cloud Deployment**: Exploring ways to deploy the app beyond our current implementation would allow us to deliver the system as per the requirements of the user.
- **User role management**: Integrating user role management would allow companies to set up hierarchies and limit access for advanced security.