1. **INTRODUCTION**
   1. **MOTIVATION:**

Managing inventory in global supply chains is a complex task influenced by factors such as fluctuating customer demand, market volatility, supplier delays, and logistical challenges. Traditional inventory systems often rely on fixed models and manual oversight, making it difficult to react quickly and accurately to these unpredictable conditions. This project is driven by the need to improve how inventory is monitored, replenished, and optimized, reducing the risk of stockouts and excess inventory.

Modern businesses are increasingly looking for ways to anticipate market behaviour and operational disruptions. By integrating data-driven insights into inventory planning, organizations can shift from reactive strategies to proactive management. This motivation stems from the idea that anticipating demand trends and potential stock issues in advance enables businesses to make better, faster decisions that directly affect customer satisfaction and operational efficiency.

Another reason for pursuing this project is the challenge of balancing inventory levels with financial efficiency. Holding too much stock increases storage costs and ties up resources, while insufficient stock can lead to missed sales and production delays. The goal is to develop a system that continuously adapts to real-time data, helping businesses maintain optimal inventory levels without the need for constant manual adjustments.

Additionally, supplier performance plays a significant role in maintaining a healthy supply chain. Variability in supplier lead times and reliability can disrupt carefully planned inventory schedules. This project is motivated by the need to monitor and assess supplier performance alongside inventory conditions, enabling businesses to manage risks better, improve operational planning, and build stronger supplier relationships.

* 1. **PROBLEM STATEMENT:**

Global supply chains often face significant challenges in managing inventory due to unpredictable demand patterns, supplier inconsistencies, and fluctuating market conditions. Traditional inventory management systems are typically rule-based and reactive, making it difficult for businesses to accurately forecast stock requirements, prevent stockouts, and avoid excess inventory. This lack of adaptability leads to operational inefficiencies, increased costs, and customer dissatisfaction.

There is a pressing need for an intelligent, data-driven approach that can predict demand, assess stockout risks, determine optimal reorder points, and evaluate supplier performance. Existing solutions either lack predictive accuracy or fail to integrate multiple factors that influence inventory decisions.

Addressing these gaps through an AI-driven system can help businesses optimize inventory levels, reduce operational risks, and make informed decisions that align with real-time supply chain dynamics.

* 1. **PROJECT OBJECTIVES:**
* To accurately forecast future demand based on historical data, seasonal trends, and market patterns.
* To predict stockout risks in advance, enabling proactive decision-making and reducing operational disruptions.
* To calculate optimal reorder points that balance stock availability with storage and procurement costs.
* To evaluate and track supplier performance, identifying delays, inconsistencies, and reliability issues.
* To provide a centralized, interactive interface for accessing real-time inventory insights and analytics.
* To improve overall supply chain efficiency, minimize inventory-related losses, and enhance customer satisfaction.



* 1. **PROJECT REPORT ORGANISATION:**

In this “Project Report,” a detailed explanation of the design challenges, proposed methodologies, and implementation of an automated system to solve the real-world problem of manual timetable creation is presented. The application's functionalities are divided into multiple modules and discussed in detail.

This report is organized into six chapters. After this introductory chapter,  
Chapter 2: It describes the literature survey, characteristics and design challenges of the existing system and provides a proposed solution.

Chapter 3: This chapter defines the software requirements, hardware requirements and user requirements.

Chapter 4: This chapter describes the use-case diagram, architecture diagram, activity diagram, sequence diagram.

Chapter 5: This chapter discusses the implementation and the testing using various tools.

Chapter 6: This chapter focuses on providing the conclusion and defines the future scope of the applications being develop

**2.LITERATURE SURVEY**

* 1. **EXISTING WORKS:**

Several existing inventory management systems have been developed over the years, ranging from conventional rule-based applications to more advanced automated platforms. Traditional methods like EOQ, JIT, and Material Requirement Planning (MRP) systems focus on minimizing costs and maintaining stock levels based on fixed formulas and demand estimates.

However, these systems often struggle to adapt to real-time changes in market conditions and global supply disruptions. In recent years, software solutions like SAP, Oracle NetSuite, and Zoho Inventory have incorporated automated features for stock tracking, order management, and supplier coordination, but many still rely heavily on user-defined parameters without predictive intelligence.

Some academic and industrial research projects have introduced machine learning-based approaches for demand forecasting and stock optimization. These include time series models, ARIMA, and decision tree-based predictions for inventory control. Although these methods have shown promise, they are often implemented in isolation, lacking integration with supplier performance metrics and risk prediction models.

* 1. **LIMITATIONS OF EXISTING WORK:**
* **Limited Use of Predictive Analytics:**

Existing systems often rely on basic algorithms or historical data without integrating advanced AI or machine learning for predictive analytics. This means they are unable to forecast demand accurately, which can result in inefficiencies like overstocking or understocking, especially during demand fluctuations.

* **Limited Flexibility and Customization:**

Many of the existing inventory management systems are rigid and don't offer enough customization to meet the specific needs of a business or industry. This limits their effectiveness, especially in industries with complex inventory requirements.

* **Poor User Interface (UI) and User Experience (UX):**

Some of the older inventory management solutions have clunky, non-intuitive interfaces that make them difficult for non-technical users to navigate. This can result in poor adoption and a steep learning curve.

* **Lack of Cloud Integration:**

Many legacy systems are not cloud-based or lack cloud integration, meaning that data management, reporting, and analysis are done locally and may be prone to data loss, security risks, or challenges with scalability.

* **Limited Analytics and Reporting:**

Existing solutions often provide basic reporting functionalities but fail to offer advanced analytics, such as demand forecasting, trends analysis, or real-time decision support. This limits the system's ability to guide strategic business decisions.

1. **Software & Hardware Specifications**
   1. **Software Requirements:**

|  |  |
| --- | --- |
| Category | Requirements |
| Web Technologies | React, HTML, CSS, FlaskAPI, Python |
| Web Browsers | Google Chrome: Version 88 or higher  - Mozilla Firefox: Version 85 or higher  - Safari: Version 14 or higher  - Microsoft Edge: Version 88 or higher |
| Operating Systems | Windows 10  - macOS 10.15 (Catalina) or higher  - Linux (e.g., Ubuntu 20.04 LTS or higher) |
| Development Tools | Visual Studio Code or any preferred code editor for web development Jupyter Notebook, Google Colab, or any Python IDE (for EDA) |
| Local Server | http |
| Version Control | Git (optional) |

**3.2 HARDWARE REQUIREMENTS**

Server Specifications

Processor: Intel Core i5 or equivalent, 2.0 GHz or higher, multi-core.

Memory: 16 GB RAM or more.

Storage: SSD with at least 500 GB storage.

Network: Gigabit Ethernet connectivity.

Database Server

Processor: Intel Core i7 or equivalent, 2.0 GHz or higher, multi-core.

Memory: 32 GB RAM or more.

Storage: SSD with at least 1 TB storage.

Network: Gigabit Ethernet connectivity.

**4. PROPOSED SYSTEM DESIGN**

* 1. **PROPOSED METHODS:**

The proposed inventory management system leverages machine learning techniques to optimize stock handling, reduce losses from overstocking or stockouts, and support data-driven decision-making. The following methods outline the approach taken across different components of the system

**1. Data Collection and Preprocessing**

* Input data is provided by the user in CSV or Excel format, including features such as:
  + Historical sales
  + SKU(Stock keeping unit) details
  + Lead times
  + Supplier information
* Preprocessing is performed to handle:
  + Missing values
  + Outliers
  + Time series alignment
  + Feature encoding and normalization (if needed)
* Data preprocessing steps were defined and refined during the Exploratory Data Analysis (EDA) phase.

**2. Demand Forecasting**

* Time series forecasting models were developed during EDA to predict future demand per product or SKU.
* Techniques used include:
  + **Prophet** for capturing seasonality and trends
  + **LSTM (Long Short-Term Memory)** networks for sequential patterns in deep learning approaches
* The trained models are reused for real-time inference, generating future demand values based on the uploaded historical data.

**3. Stockout Risk Prediction**

* A classification model was trained to predict the probability of a stockout event occurring within a specified time window.
* Model inputs include:
  + Current inventory level
  + Recent sales velocity
  + Forecasted demand
  + Lead time
* Algorithms like **Random Forest** or **XGBoost** were evaluated during EDA, with the best-performing one selected for inference.
* The output is a stockout risk score per item, guiding managers to prioritize attention.

**4. Reorder Point Calculation**

* Reorder points are calculated using demand variability and lead time data.
* The formula used is:

Reorder Point=(Average Demand×Lead Time)+Safety Stock\text{Reorder Point} = (\text{Average Demand} \times \text{Lead Time}) + \text{Safety Stock}

* Safety stock is computed based on standard deviation of demand and desired service level.
* All parameters are dynamically calculated using the data provided by the user at runtime.

**6. Output & Visualization**

* Predictions and analysis results are displayed using:
  + Line charts (for demand forecasts)
  + Bar charts or heatmaps (for stockout risk)
  + Tables and summaries (for reorder points and supplier stats)
* Users can download the results for reporting or operational planning.
* Visualization is done using **Plotly**, **Matplotlib**, or integrated components in **Streamlit** or the frontend framework.

**4.2 CLASS DIAGRAM:**

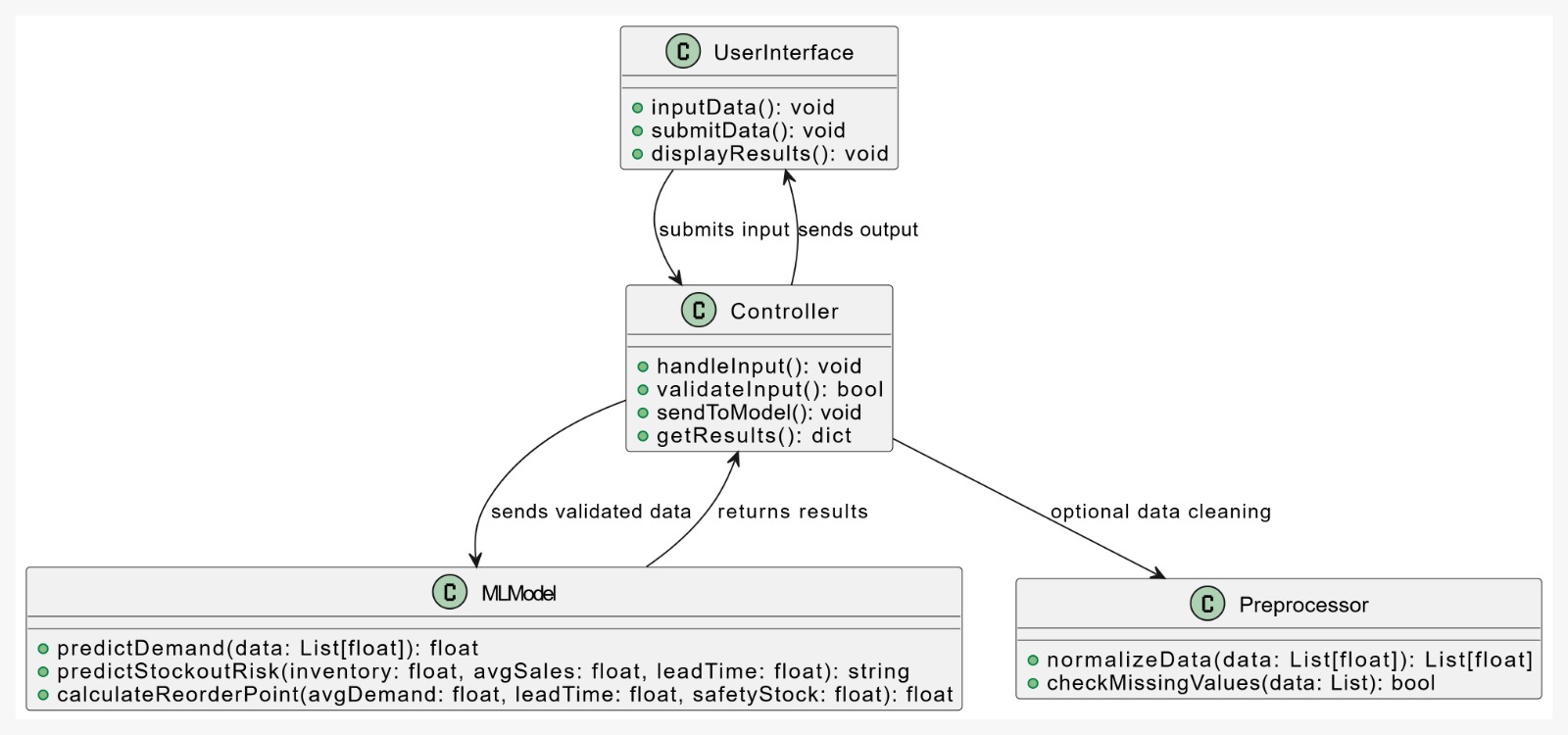


Fig 4.2.1: Class Diagram

**4.3 USE CASE DIAGRAM:**

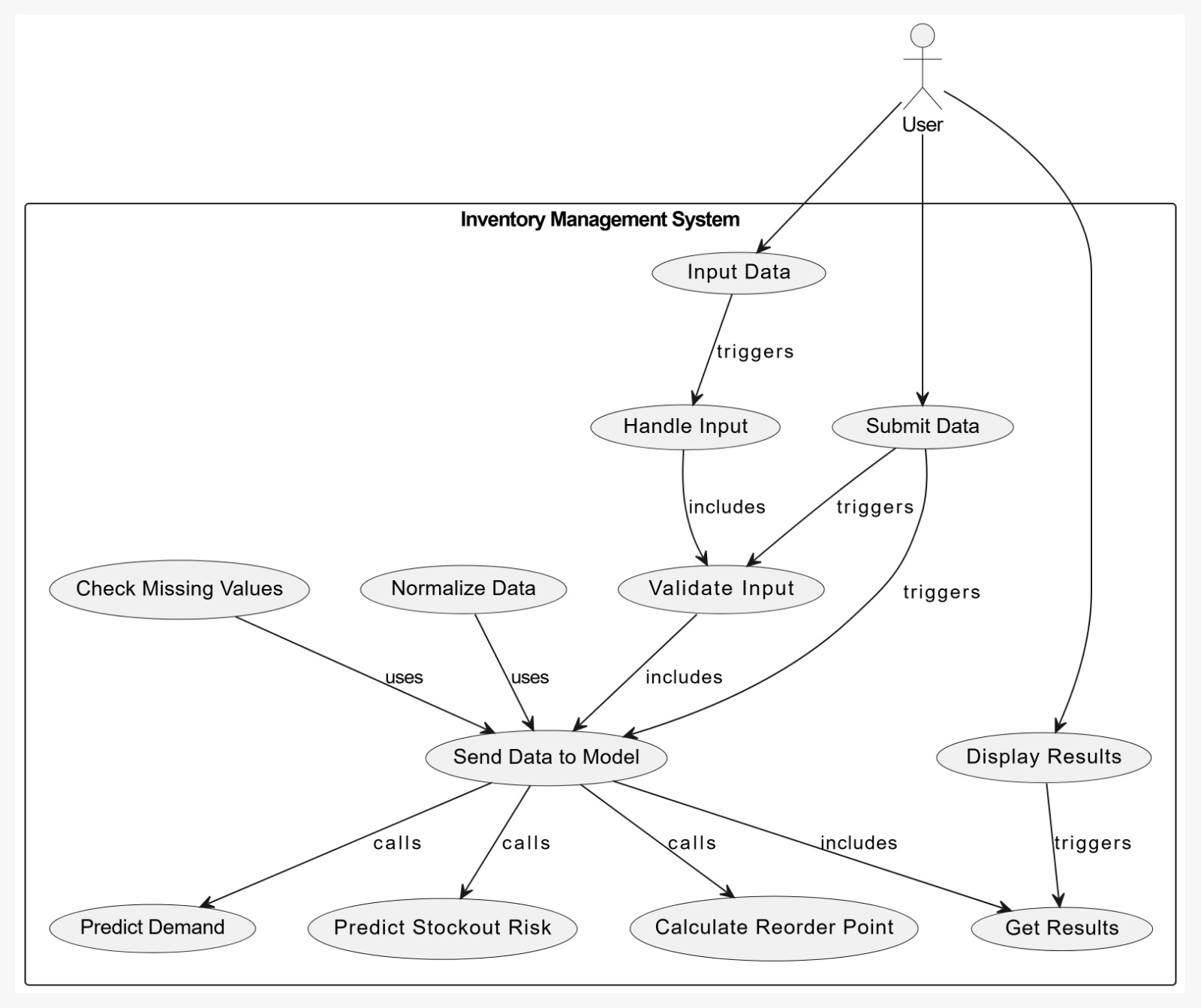
****

Fig 4.3.1: Use case Diagram

**4.4 ACTICITY DIAGRAM:**

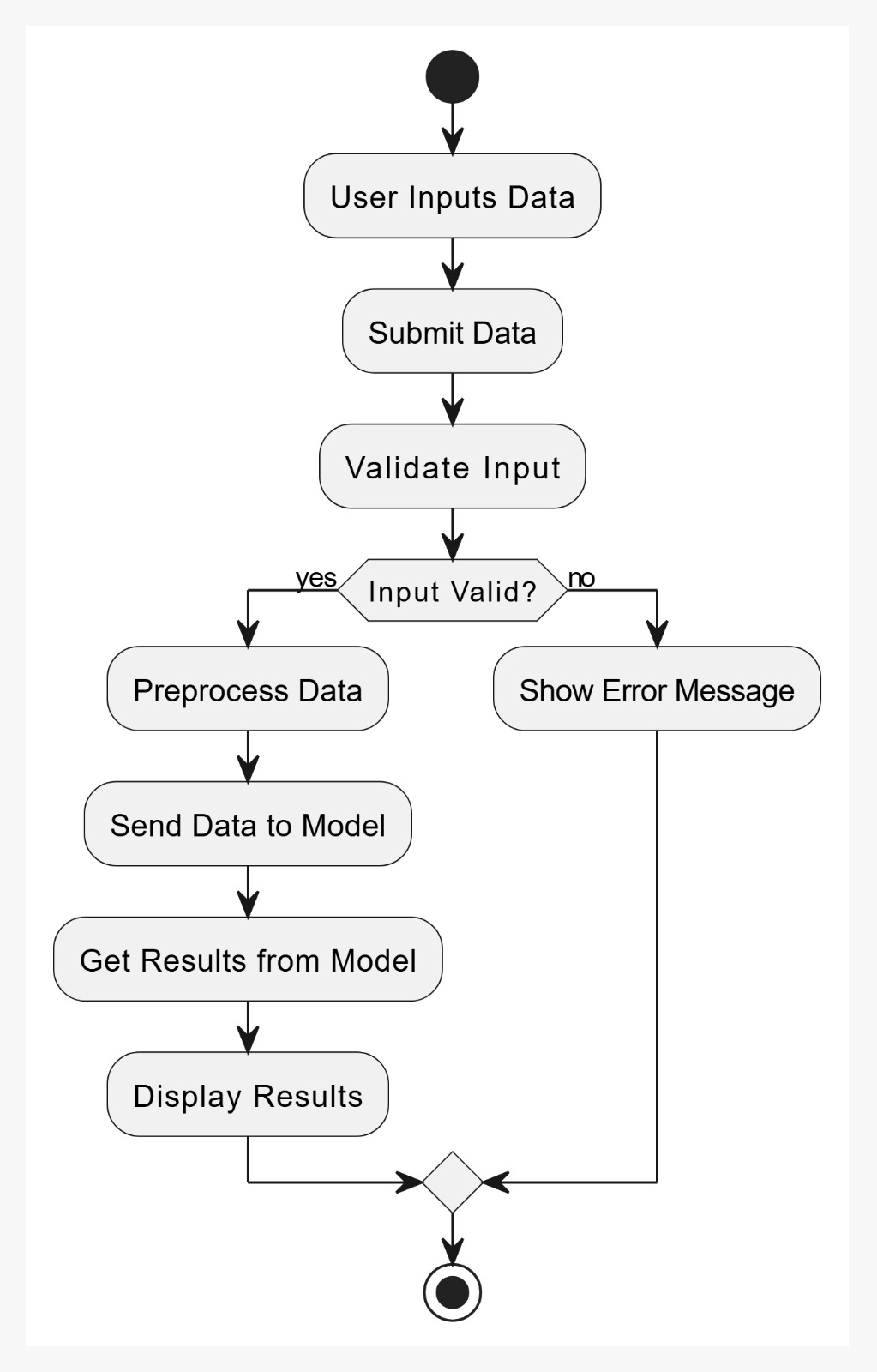
****

Fig 4.4.1: Activity Diagram

**4.5 SEQUENCE DIAGRAM:**

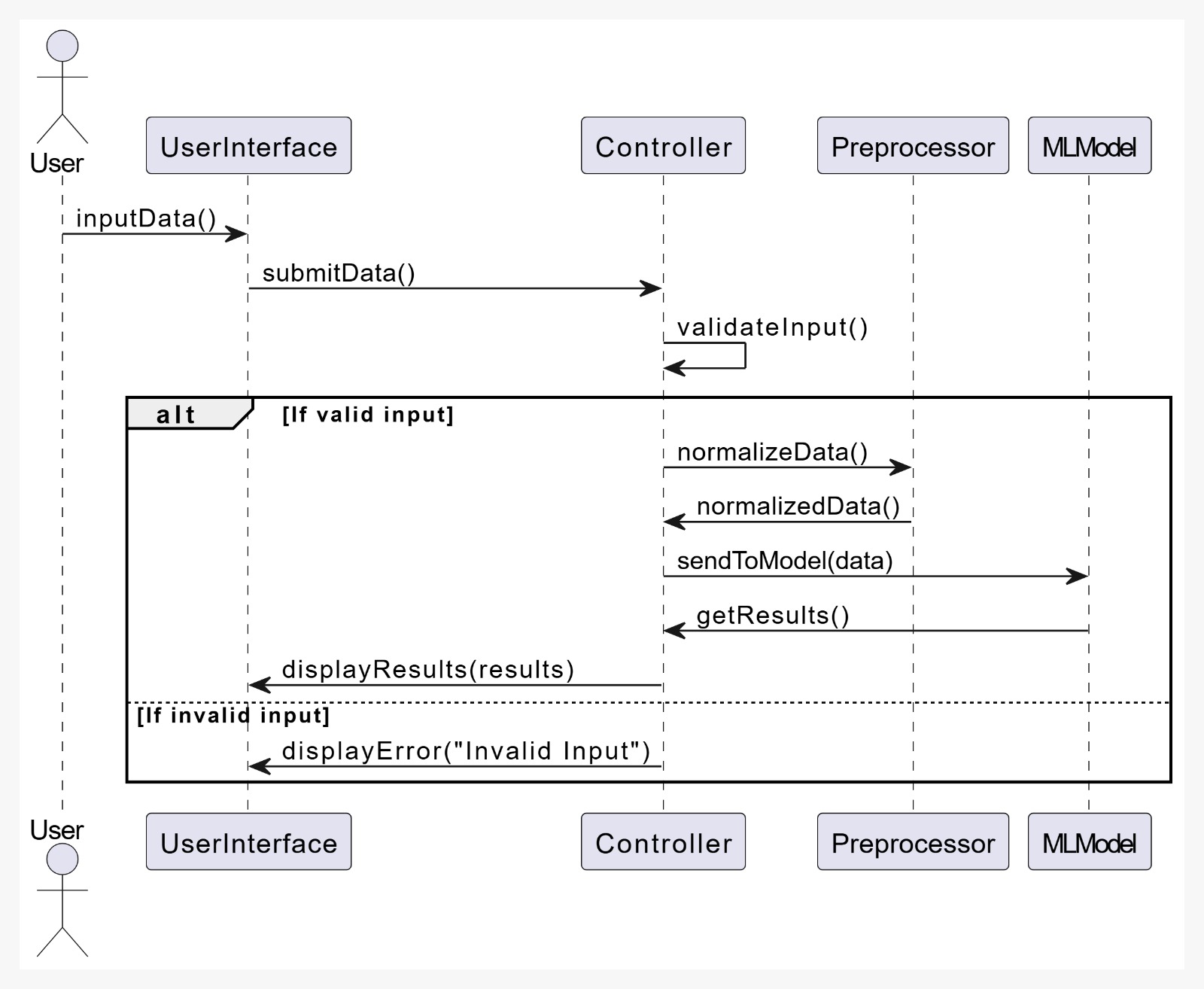
****

Fig 4.5.1: Sequence Diagram

**4.6 SYSTEM ARCHITECTURE:**

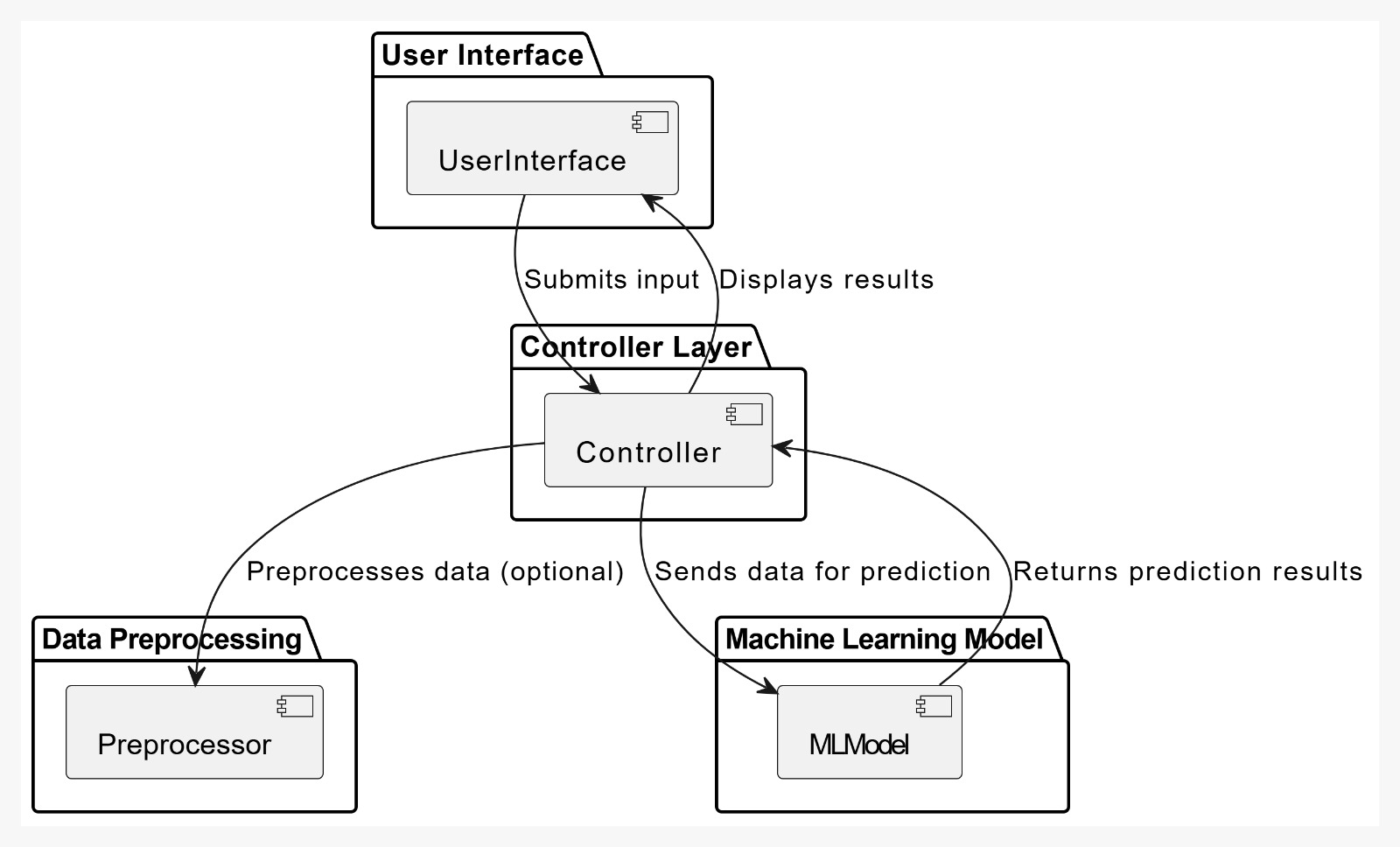
****

Fig 4.6.1: System Architecture

**4.7 TECHNOLOGY DESCRIPTION**

The following technologies are used:

**1. Flask API:**

* A lightweight and flexible web framework in Python used to build RESTful APIs.
* Manages routing, handling HTTP requests, and connecting frontend and backend.
* Supports easy integration with machine learning models and databases.

**2. Python:**

* A high-level, versatile programming language used for backend development and AI model building.
* Provides libraries like scikit-learn, pandas, and NumPy for data processing and machine learning.
* Handles core logic, data analytics, and prediction functionalities.

**3. HTML (Hypertext Markup Language):**

* The standard markup language used to structure and display content on web pages.
* Defines the layout and elements such as headings, buttons, and tables in the application’s user interface.

**4. CSS (Cascading Style Sheets):**

* A style sheet language used for designing and enhancing the appearance of HTML content.
* Controls layout, colours, fonts, spacing, and responsiveness of the web application.
* Ensures a visually appealing and user-friendly interface.

**5. IMPLEMENTATION & TESTING**

**5.1 MODULE AND FUNCTIONAL REQUIREMENTS**

**1. Module Overview**

The Inventory Management System is divided into the following functional modules:

* Demand Forecasting Module
* Reorder Point Calculator Module
* Supplier Performance Module
* Stockout Risk Prediction Module

**2. Functional Requirements**

**Demand Forecasting Module**

* Predict future product demand using machine learning algorithms.
* Analyze historical sales data, seasonality, and trends for accurate forecasting.
* Enable dynamic demand adjustments based on changing market conditions.
* Visualize demand predictions through intuitive graphs and dashboards.

**Reorder Point Calculator Module**

* Allow users to input average daily demand and lead time.
* Calculate reorder point (ROP) dynamically using the formula:  
  ROP = Average Daily Demand × Lead Time
* Automatically suggest restock alerts when inventory nears the ROP.
* Provide a user-friendly calculator interface for real-time planning.

**Supplier Performance Module**

* Track and evaluate supplier reliability based on delivery time and order accuracy.
* Maintain supplier profiles with contact details and performance metrics.
* Associate suppliers with inventory items to assess supply chain efficiency.
* Generate supplier ranking reports to optimize sourcing decisions.

**Stockout Risk Prediction Module**

* Predict the probability of stockouts using trained ML models.
* Input-based interface to assess risk for specific product IDs.
* Display risk results with indicators (e.g., Low, Medium, High).
* Help users make proactive procurement decisions based on risk level

**5.2 INTERFACES, CLASSES AND METHODS**

**Backend: Classes and Data Structures (Python)**

**1. Inventory Item (Class)**

* Represents an inventory item.
* **Attributes:**
  + item\_id — Unique identifier.
  + category — Item category.
  + quantity — Stock quantity.

**2. Supplier (Class)**

* Represents supplier information.
* **Attributes:**
  + supplier\_id — Unique identifier.

**3. Order (Class)**

* Represents stock movement records.
* **Attributes:**
  + order\_id — Unique identifier.
  + items — List of items with quantity.
  + date — Transaction date.

**API Routes (Flask API)**

**POST /orders**

* Places a new stock order.

**GET /inventory**

* Fetches the list of all items in the inventory.

**GET /orders**

* Retrieves all stock movement records.

**Frontend JavaScript Methods (React)**

**sendInventoryData(data)**

* Sends new or updated inventory data to the Flask backend via POST/PUT requests.

**fetchInventoryData()**

* Fetches inventory list from the backend and renders it on the screen.

**displayInventory(inventoryData)**

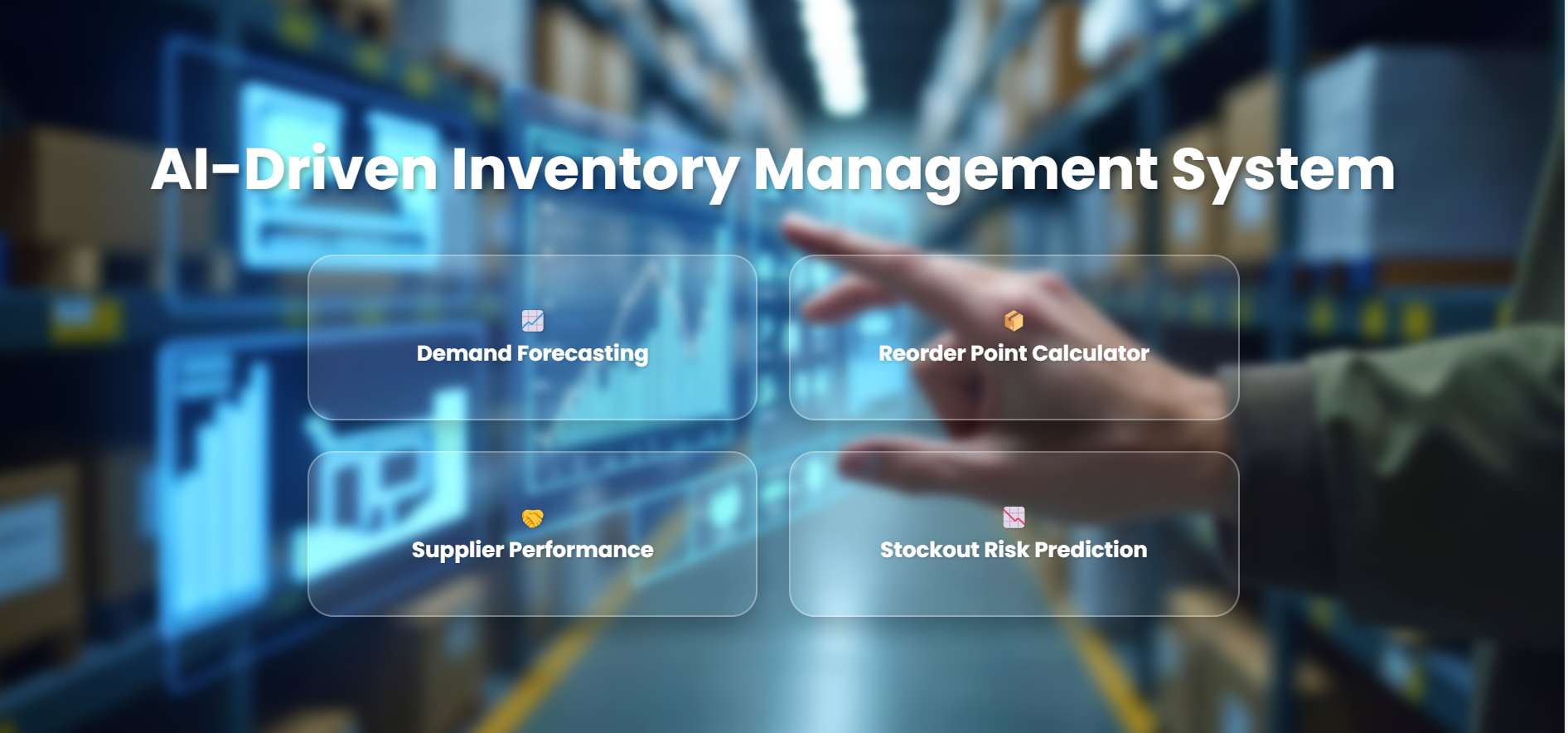
* Displays the fetched inventory in a table/grid format using React components.

**fetchOrders()**

* Retrieves order history and displays it in a list/table.

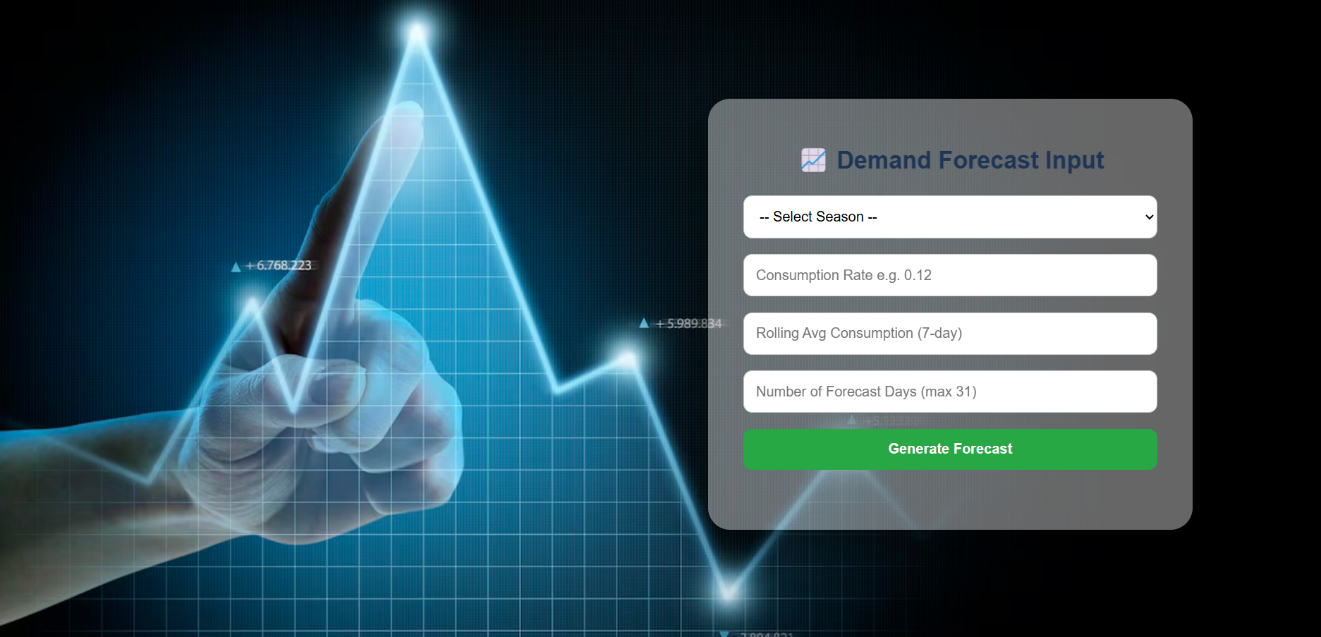
**5.3 TESTING**

This is the front page of the **AI-Driven Inventory Management System**, featuring four key modules: Demand Forecasting, Reorder Point Calculator, Supplier Performance, and Stockout Risk Prediction. It offers a clean, interactive interface for optimizing inventory operations using AI insights.

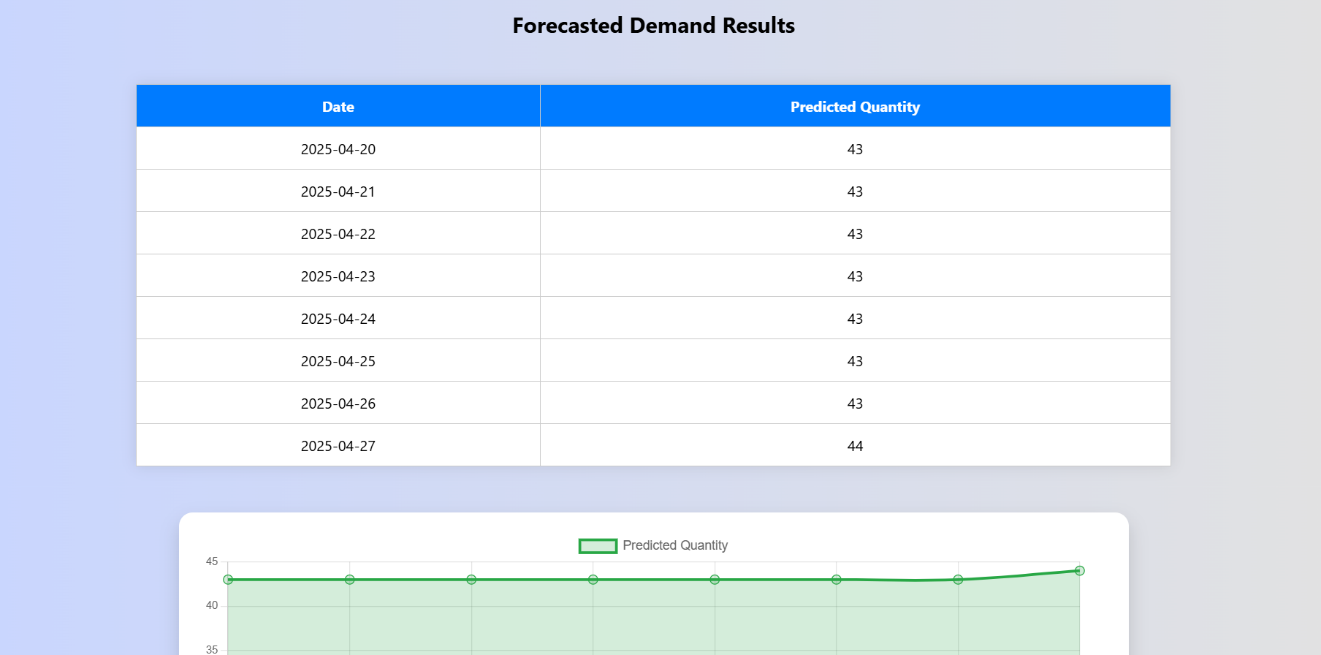


This is the **Demand Forecast Input** page it allows users to input key parameters required for generating demand forecasts. The form includes:

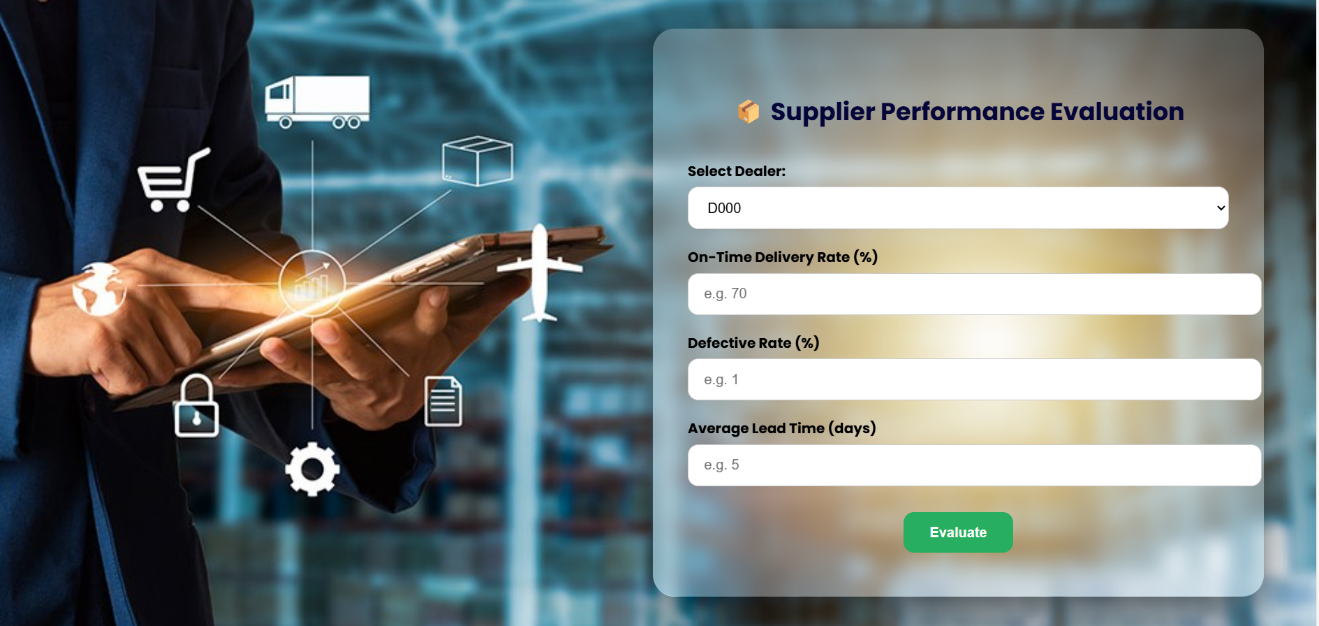
* Season Selection, Consumption Rate, Rolling Average Consumption, Forecast Period, Generate Forecast Button.



Demand Forecast are shown here



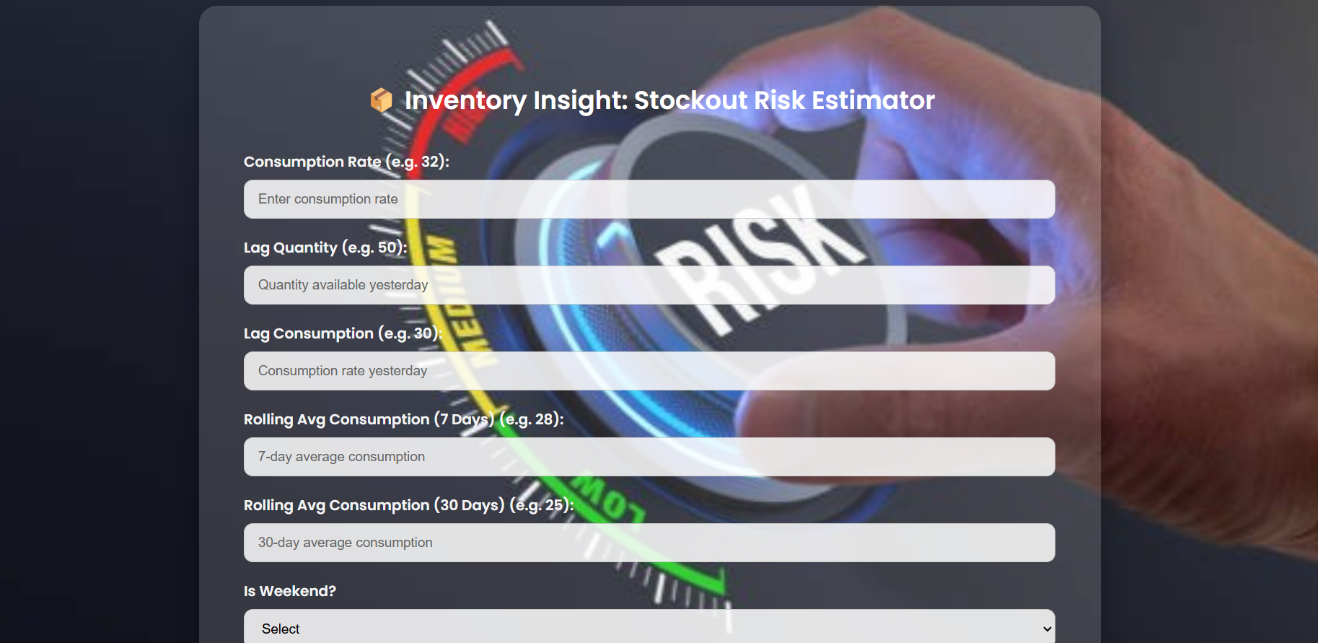
This page evaluates supplier performance based on delivery rate, defect rate, and lead time for a selected dealer. Users input values and click **Evaluate** to assess supplier efficiency.



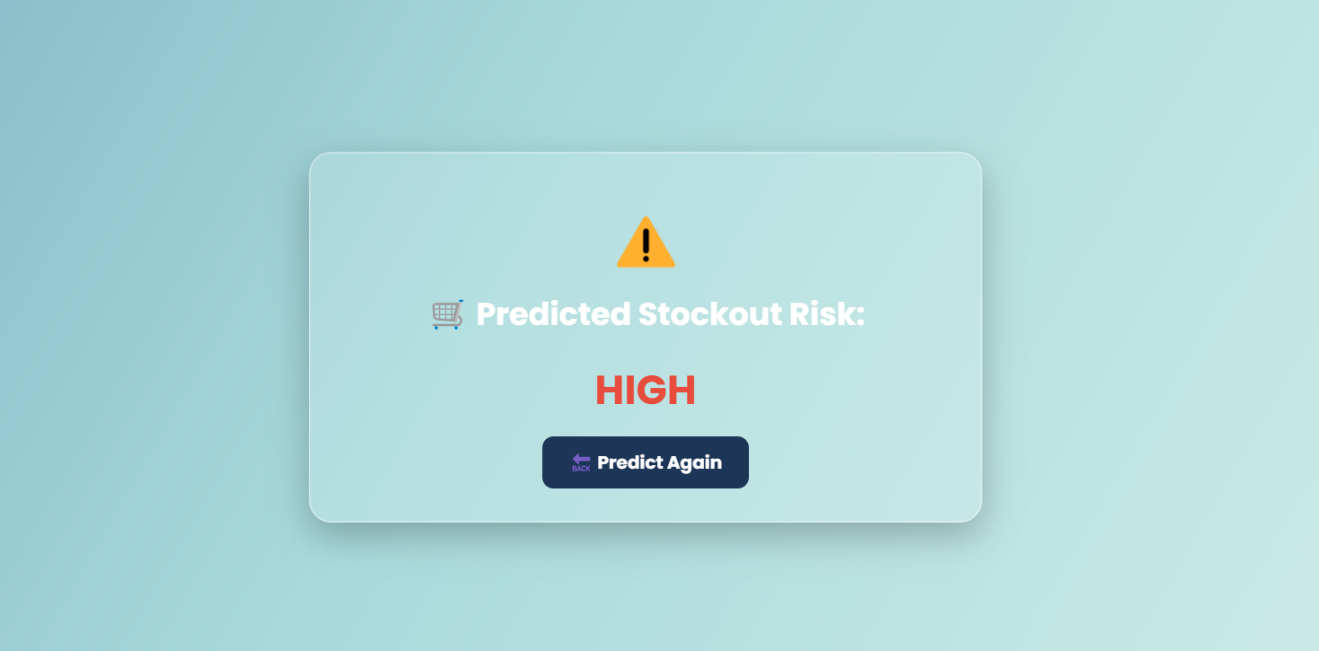
This page displays the supplier's performance result based on the evaluation inputs. It shows the dealer ID and performance status, with an option to return for re-evaluation.



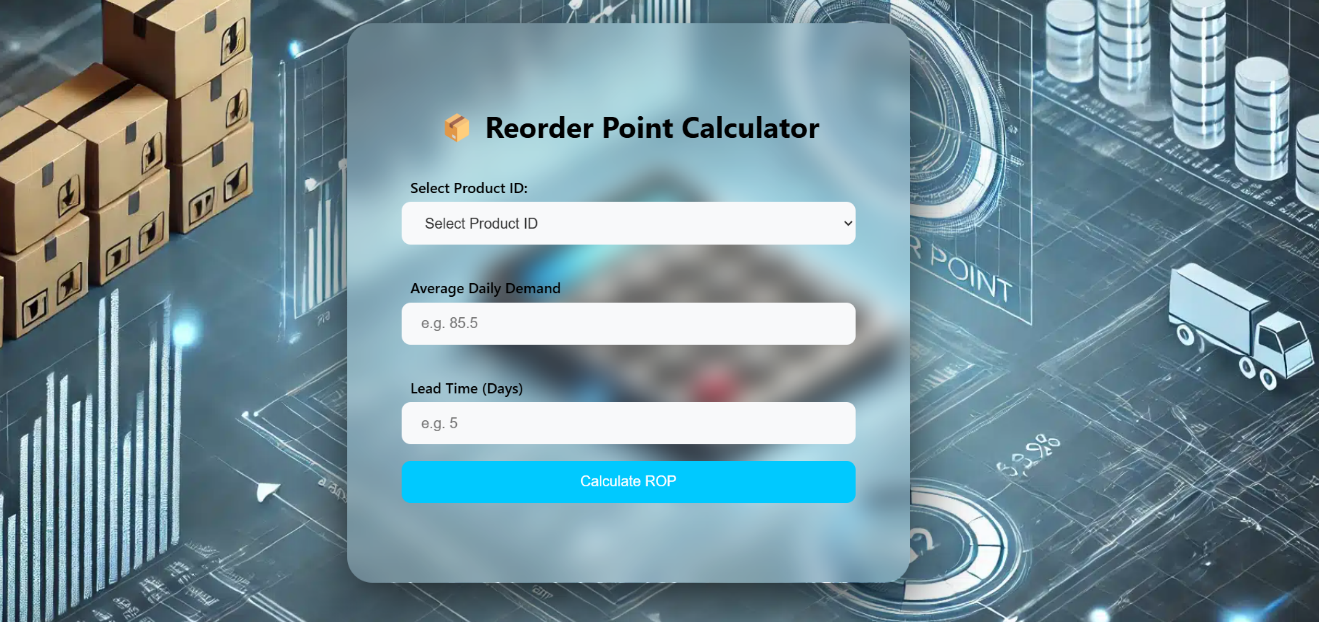
The Stockout Risk Estimator helps forecast potential inventory shortages using recent and rolling consumption trends. Input your current data to get insights and prevent supply disruptions effectively.



This page displays the estimated stockout risk level based on your input data. A "High" warning indicates an urgent need for inventory action to avoid supply chain disruption.



This calculator determines the optimal reorder point (ROP) using your product's daily demand and lead time. It helps ensure timely replenishment and minimizes stockout risks.



* 1. **CONCLUSION & FUTURE SCOPE**

**CONCLUSION:**

The **AI-Driven Inventory Management System** developed in this project leverages machine learning to optimize key areas of global supply chain management, including demand forecasting, stockout risk prediction, reorder point calculation, and supplier performance analysis. By integrating these predictive capabilities into a responsive web application built with Flask, Python, and React, the system enables businesses to make smarter, data-driven inventory decisions, reduce operational risks, and improve overall supply chain efficiency. This project demonstrates the practical impact of AI in transforming traditional inventory management processes into intelligent, adaptive systems suited for dynamic market demands.

**FUTURE SCOPE**

* **Real-time Data Integration:** Connect with IoT devices and ERP systems to enable live inventory tracking and instant decision-making.
* **Dynamic Pricing Recommendations:** Implement AI models to suggest optimal pricing based on demand, stock levels, and market trends.
* **Multi-Warehouse Optimization:** Manage inventory distribution across multiple locations efficiently, reducing logistics costs and improving service levels.
* **Automated Supplier Selection:** Use performance analytics to automatically recommend or select the most reliable and efficient suppliers.
* **Advanced Visualization Dashboards:** Integrate interactive dashboards for clearer, data-driven insights and trend analysis.
* **Mobile Application Support:** Develop a mobile-friendly version for managers and decision-makers to access insights and alerts on the move.
* **Multi-Language and Regional Customization:** Enhance global adaptability by supporting multiple languages and region-specific features.
* **Enhanced AI Models:** Continuously improve prediction accuracy and system intelligence using advanced machine learning and deep learning techniques.

**REFERENCES:**

[1] S. Shastri, A. Singh, and R. Kumar, Inventory Management Using Machine Learning. [Online].

Available: <https://ieeexplore.ieee.org/document/9186012>

[2] P. Baryannis, S. Dani, and G. Antoniou, A Survey on Supply Chain Forecasting Using Machine Learning. [Online].

Available: <https://www.sciencedirect.com/science/article/pii/S0957417421003213>

[3] D. Choi, D. Chun, and M. Lee, Machine Learning in Supply Chain Management: A Review. [Online].

Available: <https://www.mdpi.com/2071-1050/12/9/3781>

[4] G. Delen, B. Kuzgunkaya, and J. A. Cezar, Predictive Analytics for Inventory Management. [Online].

Available: <https://www.sciencedirect.com/science/article/pii/S0360835218301241>

[5] R. H. Ahmed, A. T. Azad, and A. Rahman, Demand Forecasting with Machine Learning Models in Retail. [Online].

Available: <https://www.frontiersin.org/articles/10.3389/frai.2020.00028/full>