22.11.2024

TED UNIVERSITY

CMPE 491

Senior Project I

Project Analysis Report: AI-Powered Inventory and Quality Monitoring System

Project Team: Erdem Atak, Zeynep Bakanoğulları, İrem Su Gül

Advisor: Tansel Dökeroğlu

Jury Members: Eren Ulu, Fırat Akba

Table of Contents

1. Introduction

The AI-Powered Inventory and Quality Monitoring System represents an innovative solution designed to address critical challenges in inventory management and quality control, particularly for businesses handling perishable goods. This analysis document examines the system requirements, constraints, and proposed solutions based on the initial project specifications.

The system aims to revolutionize traditional inventory management by incorporating advanced technologies such as computer vision and machine learning to automate various aspects of stock tracking and quality control. This analysis will detail how the system will function, its requirements, and the challenges it aims to address.

2. Current System

Currently, most businesses rely on manual processes for inventory management and quality control, which presents several challenges:

- Manual stock tracking leading to human error
- Inefficient quality monitoring processes
- Delayed detection of product deterioration
- Manual expiration date checking
- Reactive rather than predictive inventory management
- Time-consuming physical inventory counts
- Inconsistent quality assessment standards
- Limited data collection for trend analysis
- High dependency on human judgment for quality assessment

These limitations result in:

- Increased product waste
- Higher operational costs
- Reduced efficiency
- Inconsistent quality control
- Potential customer dissatisfaction
- Financial losses due to spoilage
- Inefficient resource allocation

3. Proposed System

3.1 Overview

The proposed system automates inventory management and quality monitoring using image recognition and machine learning models. Key modules include stock tracking, quality assessment, expiration monitoring, and inventory forecasting. The system ensures accuracy, reduces manual intervention, and optimizes stock management decisions.The proposed system will integrate several key technologies to create a comprehensive solution:

1. **Quality Control System**
    - Computer vision-based defect detection
    - Automated updates to reflect changes in the inventory
    - Visual inspection using image recognition to detect defects.
    - Assessing product deterioration levels through machine learning
    - Classification of product conditions (e.g., normal, deteriorated, defective)
2. **Stock Tracking System**
    - Real-time monitoring of stock levels
    - Integration with existing inventory databases
    - Continuous stock level updates
3. **Expiration Management**
    - Automated date recognition
    - Proactive expiration alerts
    - Systematic tracking
4. **Inventory Forecasting System**
    - Time-series sales data analysis
    - Predictive inventory management
    - Forecasting future stock requirements to avoid under- or over-stocking.

5. **User Interface System**
    - A web-based dashboard providing:

- Summary of inventory levels.
- Quality alerts and forecasts.
- Detailed reports.

3.2 Functional Requirements

3.2.1 Stock Tracking Module

1. Real-time Inventory Monitoring
    - Automated stock level tracking
    - Product identification and counting
    - Stock movement tracking
    - Inventory reconciliation
    - Alert generation for low stock levels
2. Database Integration
    - Real-time synchronization
    - Historical data maintenance
    - Data backup and recovery

3.2.2 Quality Monitoring Module

1. Defect Detection
    - Visual analysis of products
    - Automated classification
    - Defect categorization
2. Image Processing
    - Processing of the image to be compatible with machine learning model
    - Multiple angle analysis if the images are available
    - Image storage

3.2.3 Deterioration Level Tracking

1. ML-Based Model
    - Monitoring of detention based on images
    - Assessing product deterioration levels through machine learning.
    - Progressive deterioration tracking
2. Alert System
    - Generating notifications on the system nearing critical detonation levels detected by ML model
    - Automated reporting

3.2.4 Expiration Date Monitoring

1. Date Recognition
    - OCR-based date extraction
    - Format standardization
    - Batch processing capability
    - Verification system
2. Alert Management
    - Automated alerts for products nearing expiration.
    - Reporting features for proactive decision-making.
    - Action recommendation

3.2.5 Inventory Forecasting System

1. Data Analysis
    - Historical sales analysis
    - Seasonal trend identification
    - Demand pattern recognition
2. Prediction System
    - Stock level optimization
    - Forecasting future stock requirements to avoid under- or over-stocking

3.3 Nonfunctional Requirements

3.3.1 Performance

- System must process single product image analysis within 10 seconds
- Dashboard must load within 1.5 seconds with up to 10,000 inventory items
- System must support minimum 100 concurrent users without performance degradation
- Database queries must complete within 5 seconds for standard operations
- Image processing pipeline must handle minimum 1 images per minute
- System uptime of 99.9% measured monthly (maximum 43.2 minutes downtime per month)
- Maximum memory usage must not exceed 8GB under full load
- API endpoints must respond within 200ms for GET requests and 500ms for POST requests

3.3.2 Security

- Implementation of AES-256 encryption for data at rest
- Password requirements: minimum 6 characters, must include uppercase, lowercase, numbers, and special characters
- Failed login attempts limited to 5 within 15 minutes before temporary account lockout
- All system actions must be logged with timestamp, user ID, and IP address
- Database backups every 6 hours with 30-day retention period

3.3.3 Reliability

- System recovery time (RTO) must not exceed 4 hours
- Automated failover mechanism must activate within 30 seconds of primary system failure
- Data consistency checks must run every 15 minutes
- Error logs must be maintained for minimum 90 days
- System must handle minimum 10,000 transactions per hour
- Automated backup system with:
    - Hourly incremental backups
    - Daily full backups
    - Weekly off-site backups
    - Monthly archive backups with 1-year retention
- Error reporting within a day

3.3.4 Usability

- Maximum 5 clicks to reach any major function from the main dashboard
- All critical functions must be accessible via keyboard shortcuts
- System response feedback within 500ms of user action
- Error messages must be specific and suggest corrective action
- Search results must appear within 1 seconds
- UI must support minimum screen resolution of 1366x768
- Form completion time must not exceed 1 minute for standard operations
- System must support the following browsers:
    - Chrome
    - Firefox
    - Safari
    - Edge
- Interface text must maintain minimum contrast ratio of 4.5:1
- All interactive elements must have minimum touch target size of 44x44 pixels

3.3.5 Maintainability

- Code documentation coverage minimum 80%
- Automated test coverage minimum 85%
- Maximum bug resolution time:
    - Critical: 4 hours
    - High: 24 hours
    - Medium: 72 hours
    - Low: 1 week
- System logs must be structured in JSON format
- API versioning system with support for minimum 2 previous versions

3.3.6 Scalability

- System must scale horizontally to handle 100% increase in load
- Database must support minimum 100GB of data with query performance degradation not exceeding 10%
- Storage system must support 50% annual growth in data volume
- Message queue must handle minimum 1000 messages per second

3.3.7 Compatibility

- REST API must support JSON format
- Data export in CSV, JSON, and PDF formats based on the user need

- Support for standard image formats:
  - JPEG (up to 10MB)
  - PNG (up to 20MB)

- Database compatibility with PostgreSQL 13

3.4 Pseudo Requirements

1. **Hardware Requirements**:
   - Cameras for image capture from different angles.
   - Sufficient storage for historical data.
2. **Employee Adaptation**:
   - User-friendly interfaces to minimize resistance from staff to learn the system .
   - Training sessions to ensure easy and smooth adoption.

3. Ethical Requirements

- ACM code of ethics and IEEE code of ethics will be followed.

3.5 System Models

3.5.1 Scenarios

Scenario 1: Inventory Stock Update

- **Actor:** Inventory Manager
- **Goal:** Update stock levels after a new shipment arrives.
- **Preconditions:** The shipment has been logged in the system, and images of products are available.
- **Steps:**
    1. Inventory Manager logs into the system via the dashboard.
    2. Selects "Stock Tracking" from the menu.
    3. Uploads product images via the stock update interface.
    4. System identifies and categorizes products using image recognition.
    5. Updated stock levels are reflected in the dashboard.
- **Postconditions:** Stock levels are updated, and alerts for critical stock are resolved if applicable.

Scenario 2: Quality Check Alert

- **Actor:** Quality Control Specialist
- **Goal:** Identify and address defective products.
- **Preconditions:** System has processed images of products and flagged defects.
- **Steps:**
    1. Quality Control Specialist accesses the "Quality Alerts" section of the dashboard.
    2. Views flagged products and defect details (e.g., type and location of defect).
    3. Marks items for removal or sends them for further inspection.
    4. Updates the status in the system to reflect the action taken.
- **Postconditions:** Defective products are removed or addressed, and the system records the action.

Scenario 3: Expiration Date Notification

- **Actor:** Store Supervisor
- **Goal:** Take proactive actions on products nearing expiration.

- **Preconditions:** System has scanned expiration dates and flagged items nearing their expiration.
- **Steps:**
    1. Supervisor receives an automated expiration alert on the dashboard.
    2. Navigates to the "Expiration Management" module.
    3. Reviews the list of products nearing expiration and associated details.
    4. Decides on actions such as discounts or removal from inventory.
    5. Updates product status in the system.
- **Postconditions:** Products nearing expiration are managed appropriately, minimizing waste.

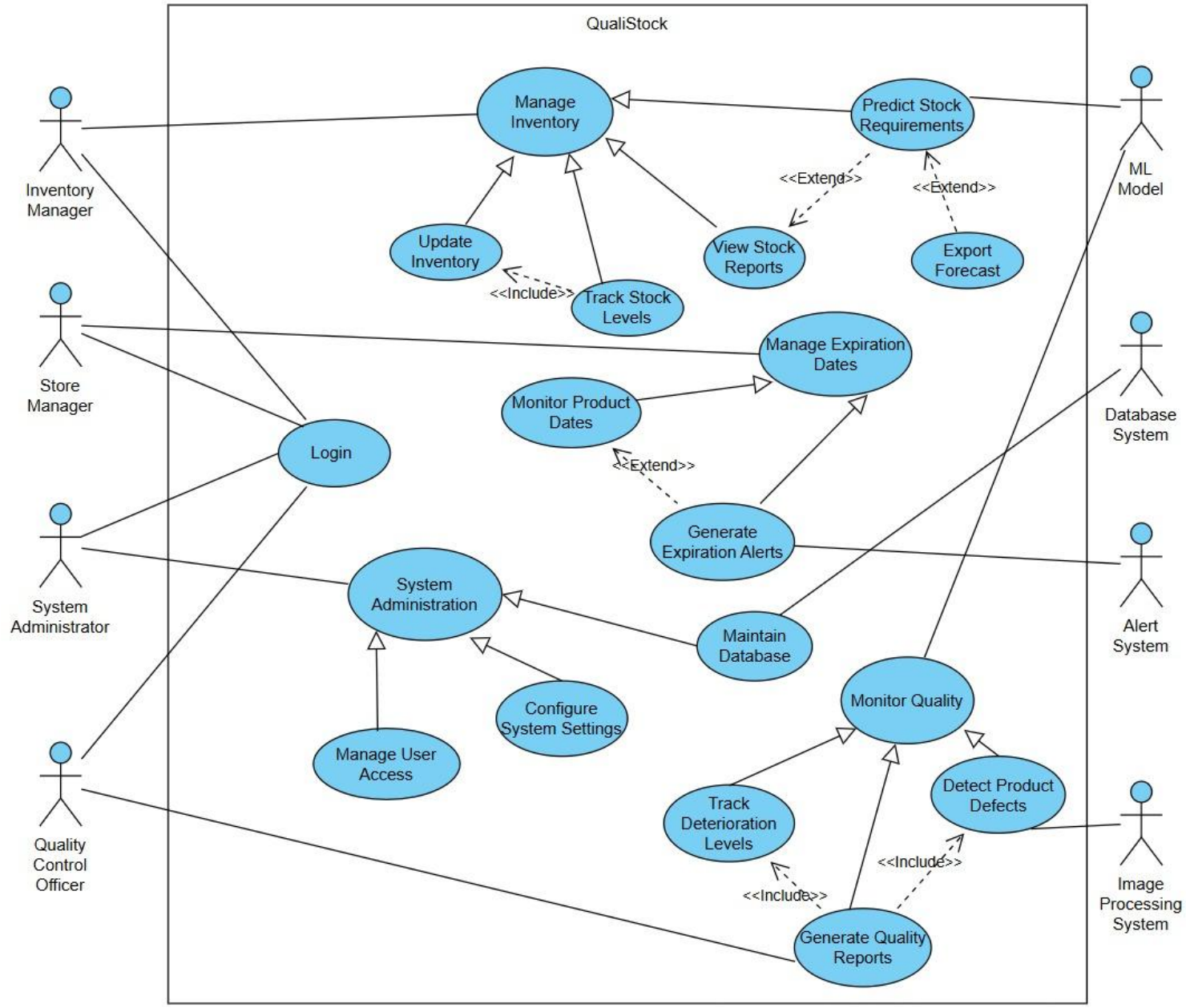
Scenario 4: Forecasting Inventory Needs

- **Actor:** Supply Chain Analyst
- **Goal:** Plan inventory orders based on predictive data.
- **Preconditions:** Historical sales data and seasonal trends are available.
- **Steps:**
    1. Analyst accesses the "Forecast Insights" section on the dashboard.
    2. Views demand forecasts for the upcoming weeks or months.
    3. Downloads reports with recommendations for reorder quantities.
    4. Communicates recommendations to procurement teams.
- **Postconditions:** Inventory levels are optimized to meet future demand.

Scenario 5: System Downtime and Recovery

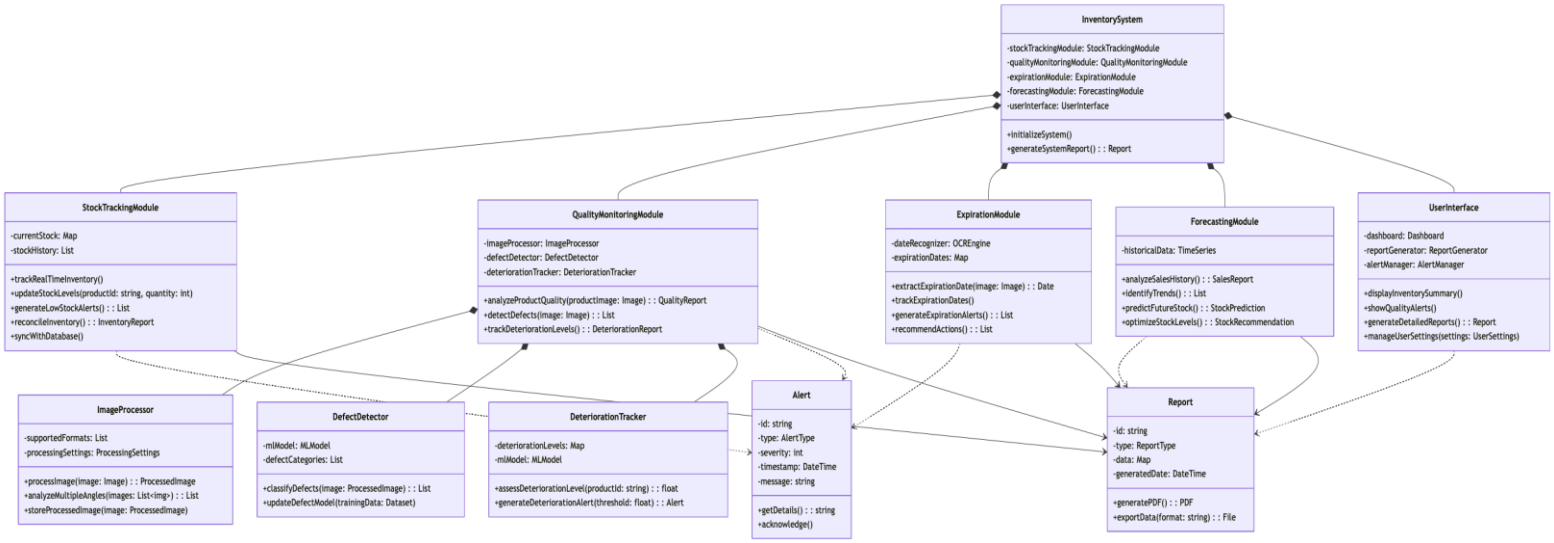
- **Actor:** System Administrator
- **Goal:** Ensure minimal disruption during downtime.
- **Preconditions:** System experiences an unexpected failure.
- **Steps:**
    1. Administrator is alerted by the automated failover mechanism.
    2. Logs into the backup system to monitor ongoing operations.
    3. Investigates root cause using error logs.
    4. Restores the primary system within the specified time.
- **Postconditions:** System returns to normal operations with minimal downtime and data loss.

3.5.2 Use Case Diagram



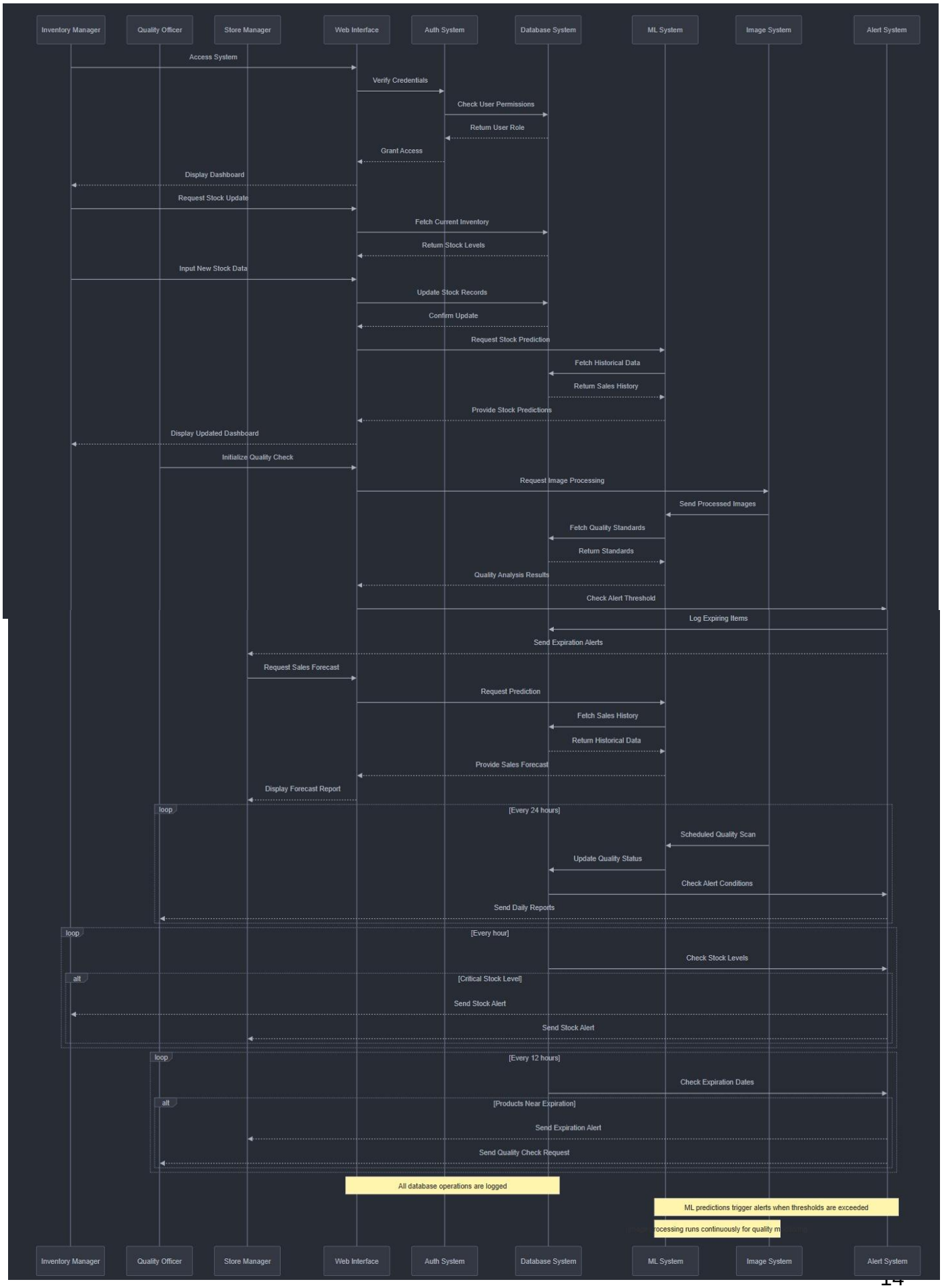
Use Case Diagram

3.5.3 Class Diagram



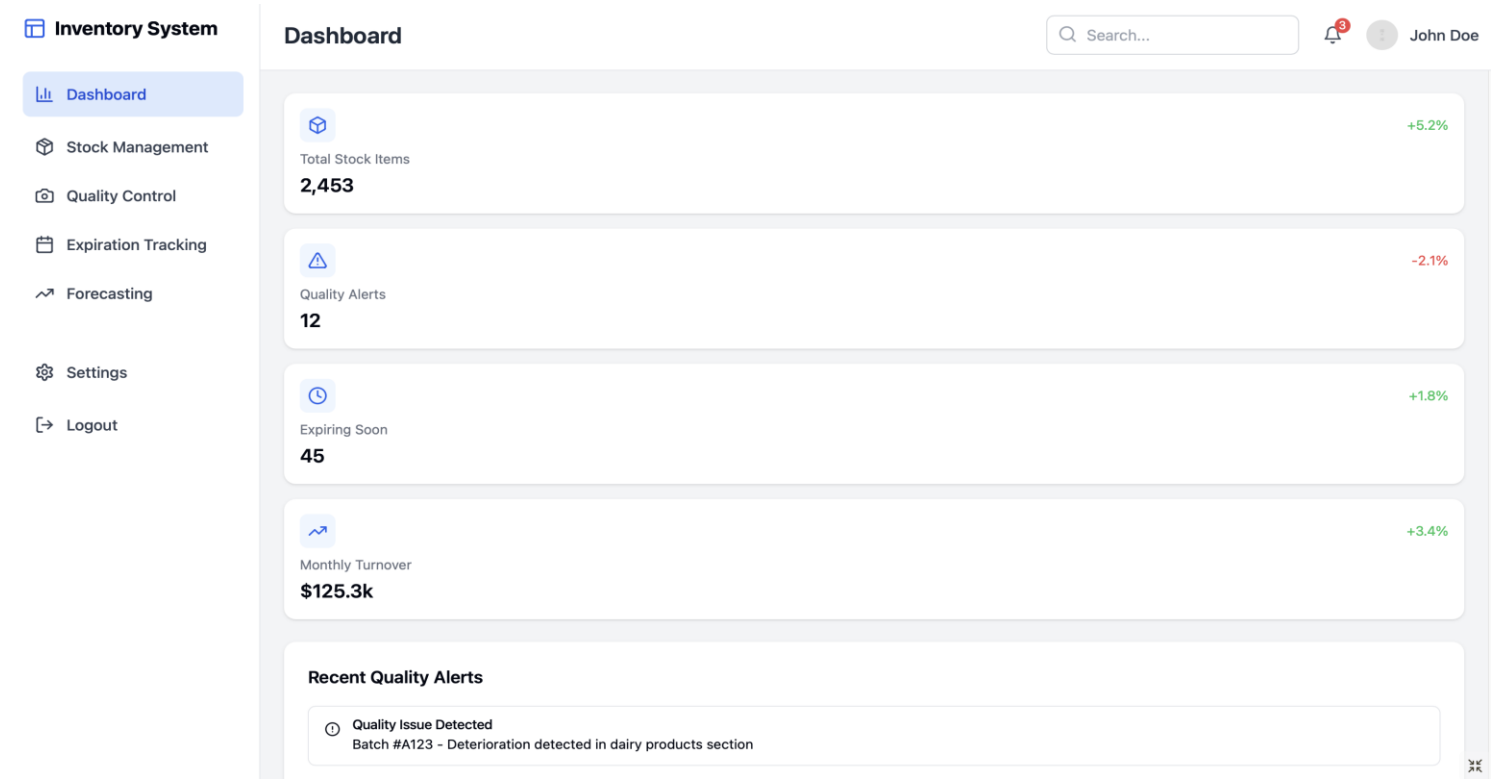
Class Diagram

3.5.4 Dynamic Models

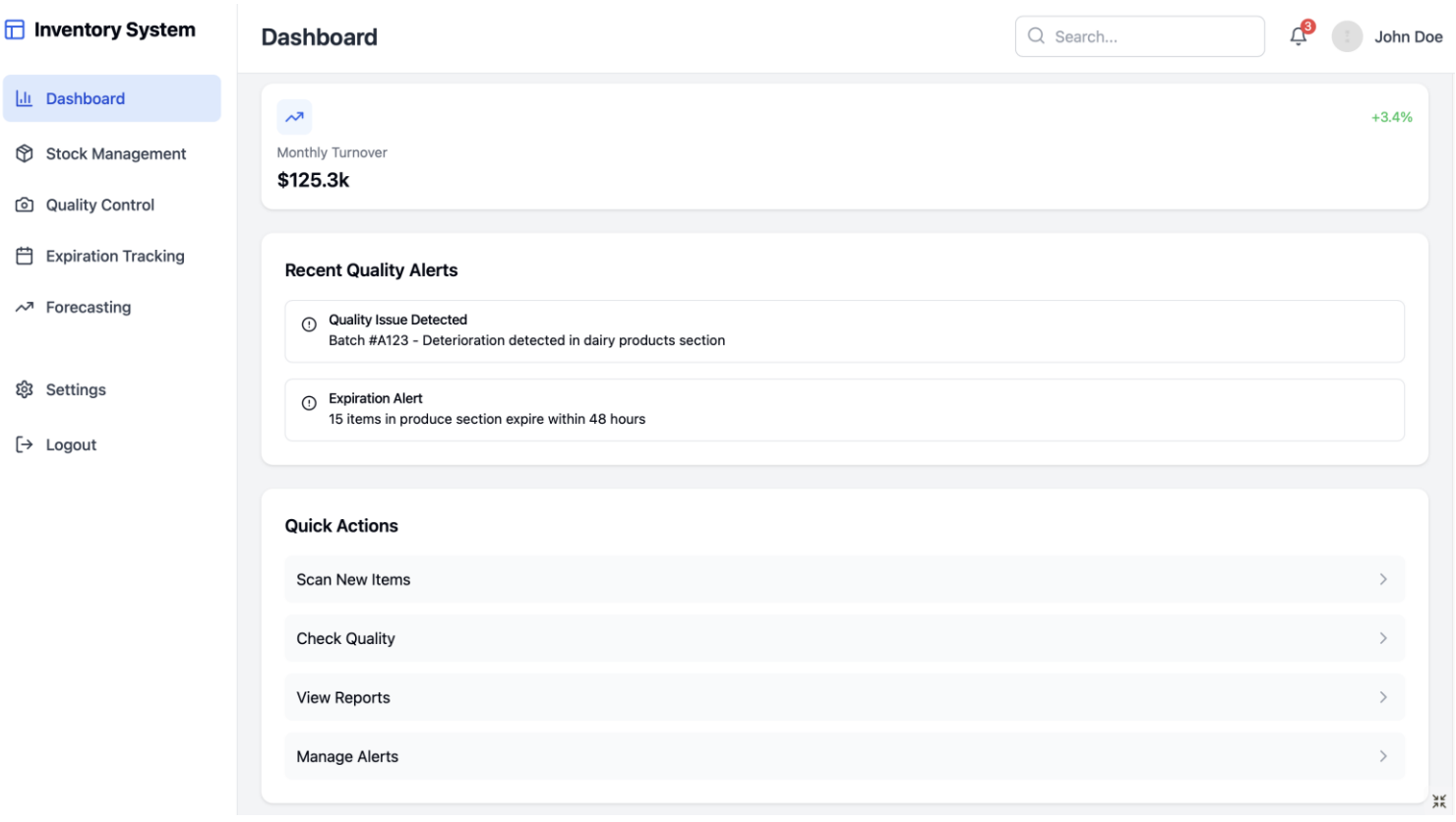


Sequence Diagram 2

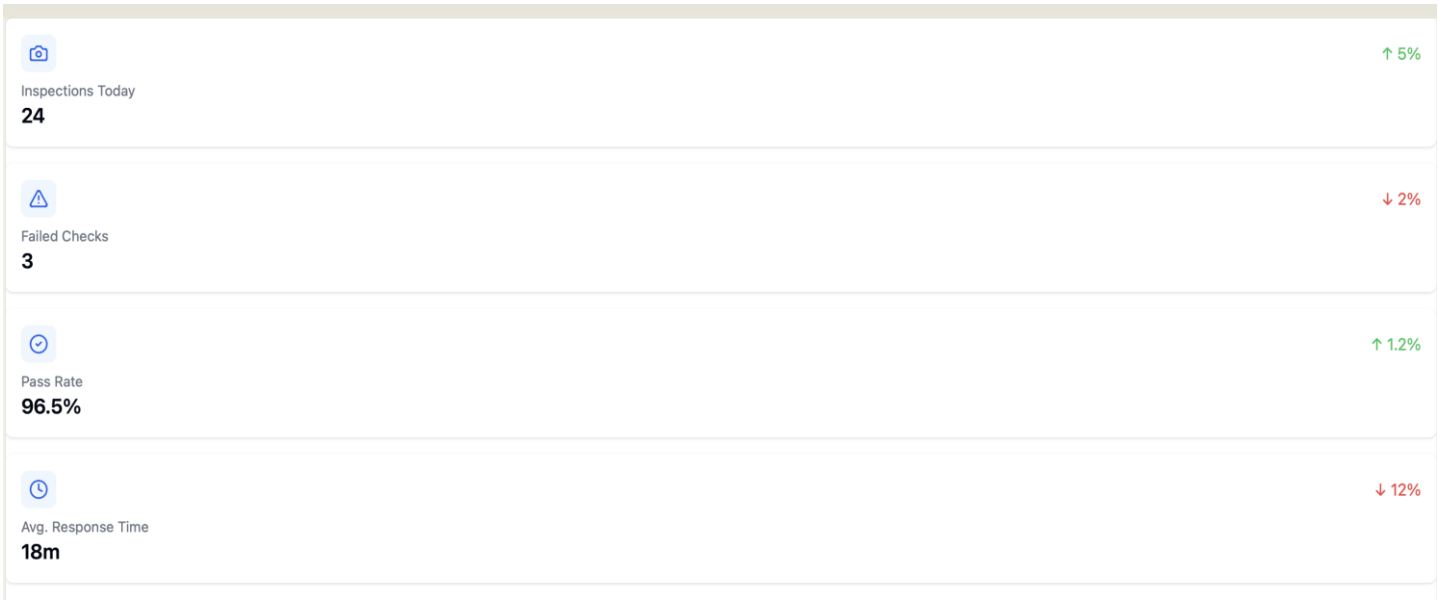
3.5.5 User Interface - Navigational Paths and Screen Mock-ups



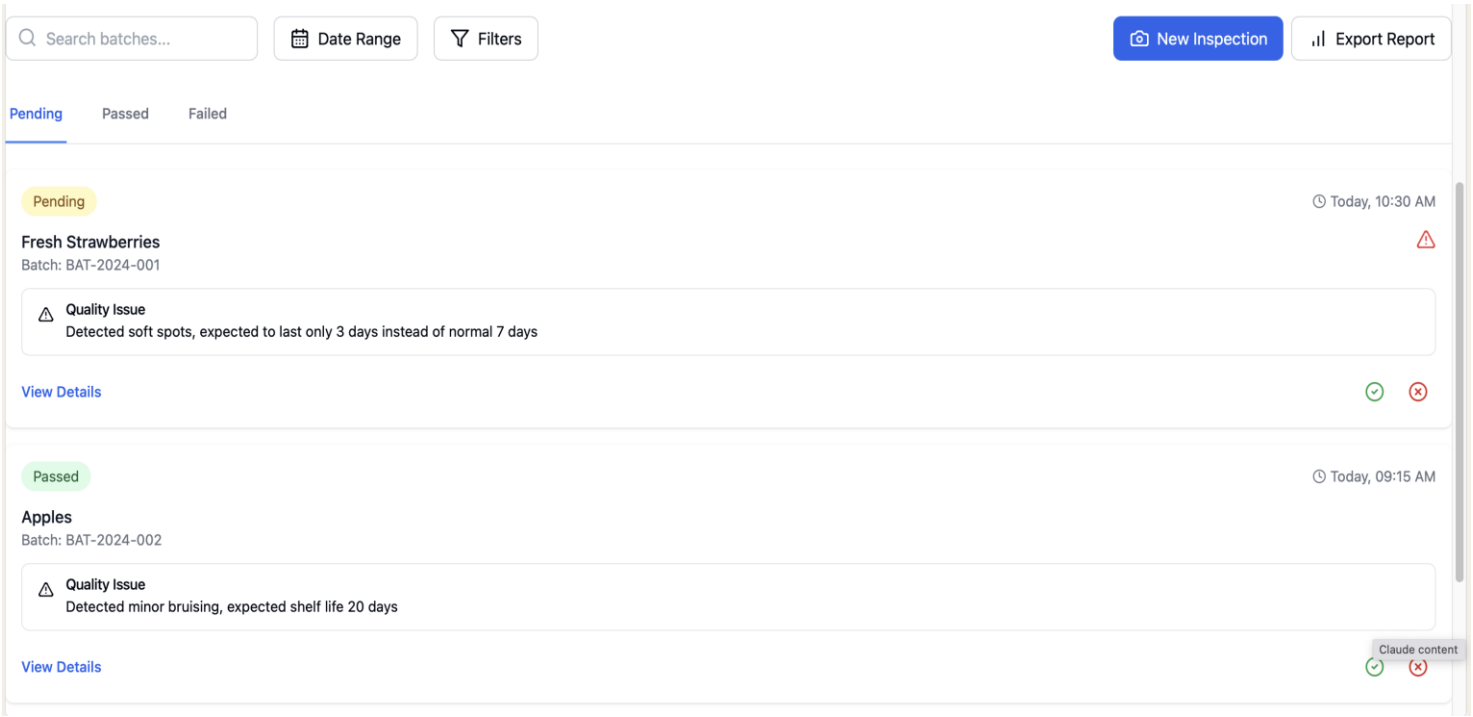
Main Dashboard 1



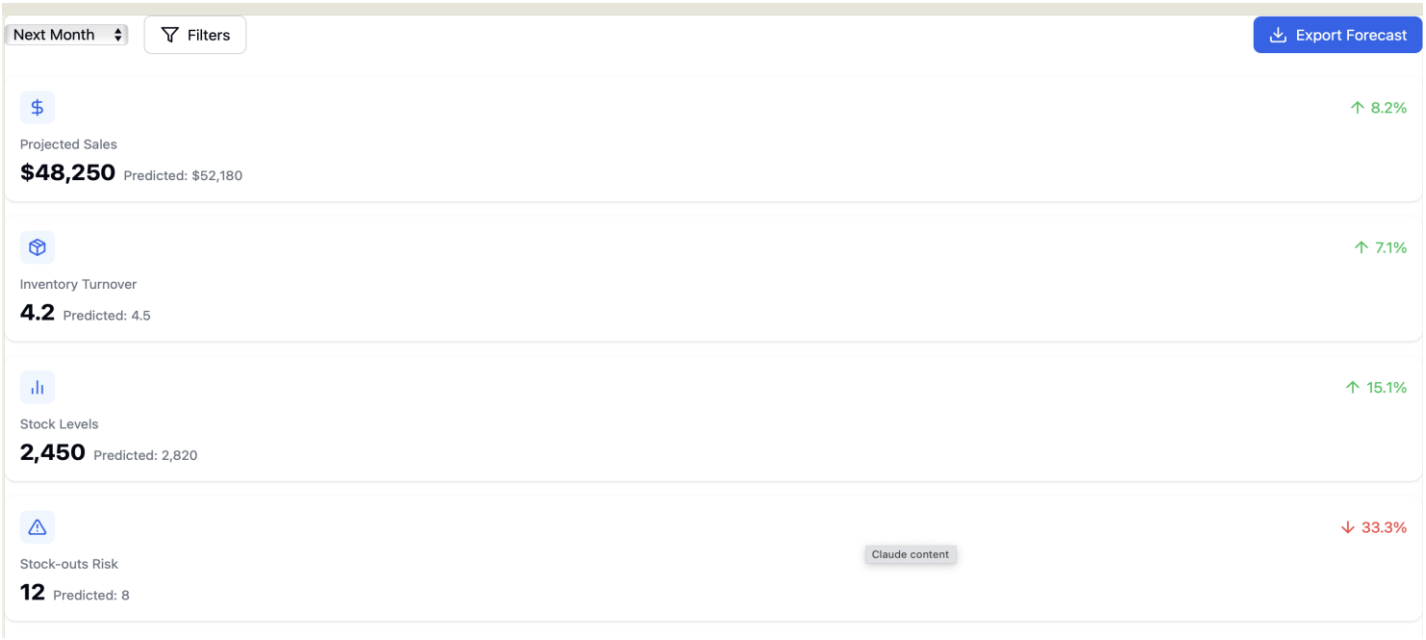
Main Dashboard 2



Quality Control 1

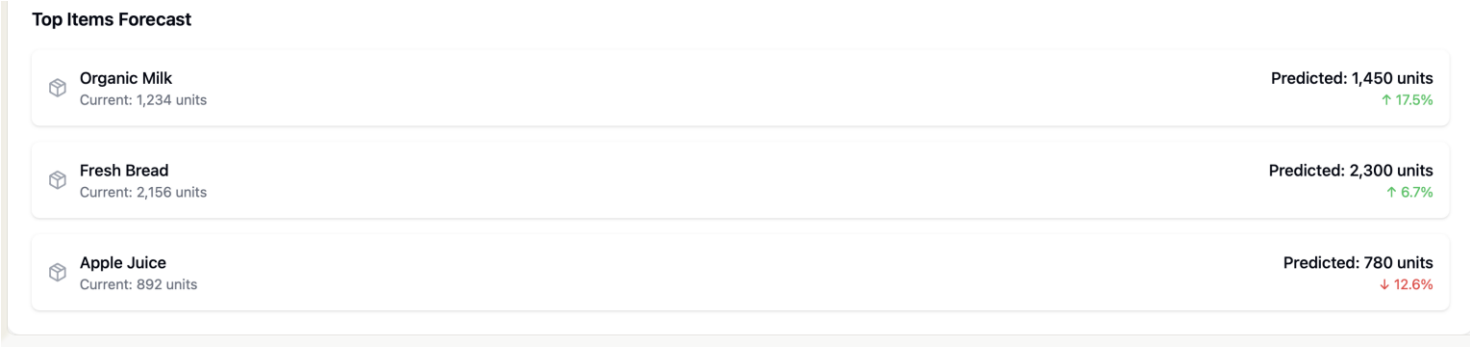


Quality Control 2

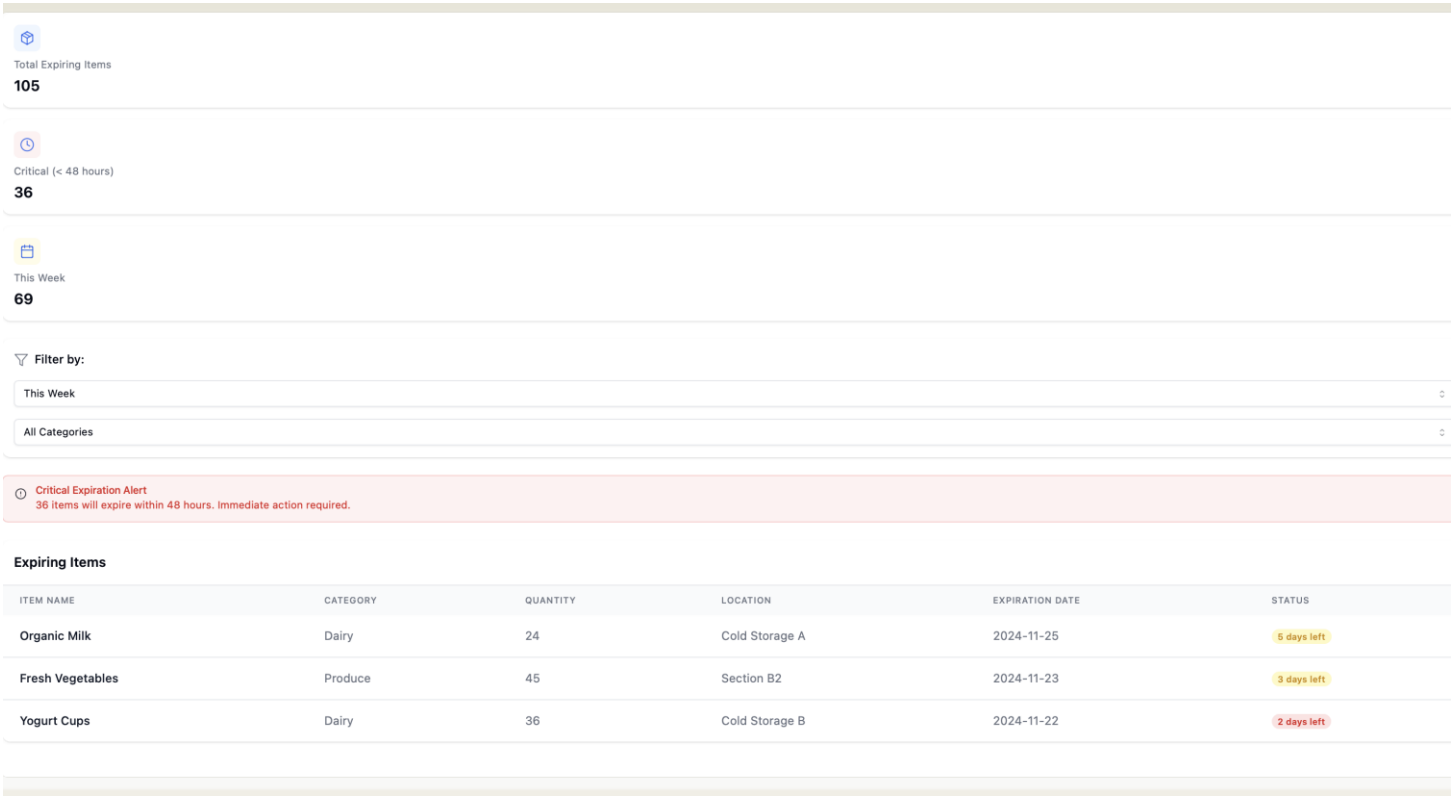


Forecast 1

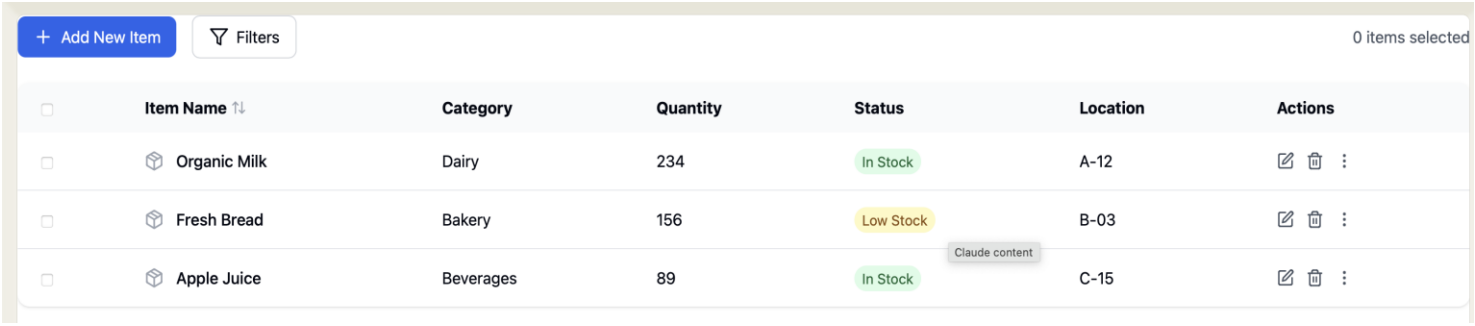
Forecast 2



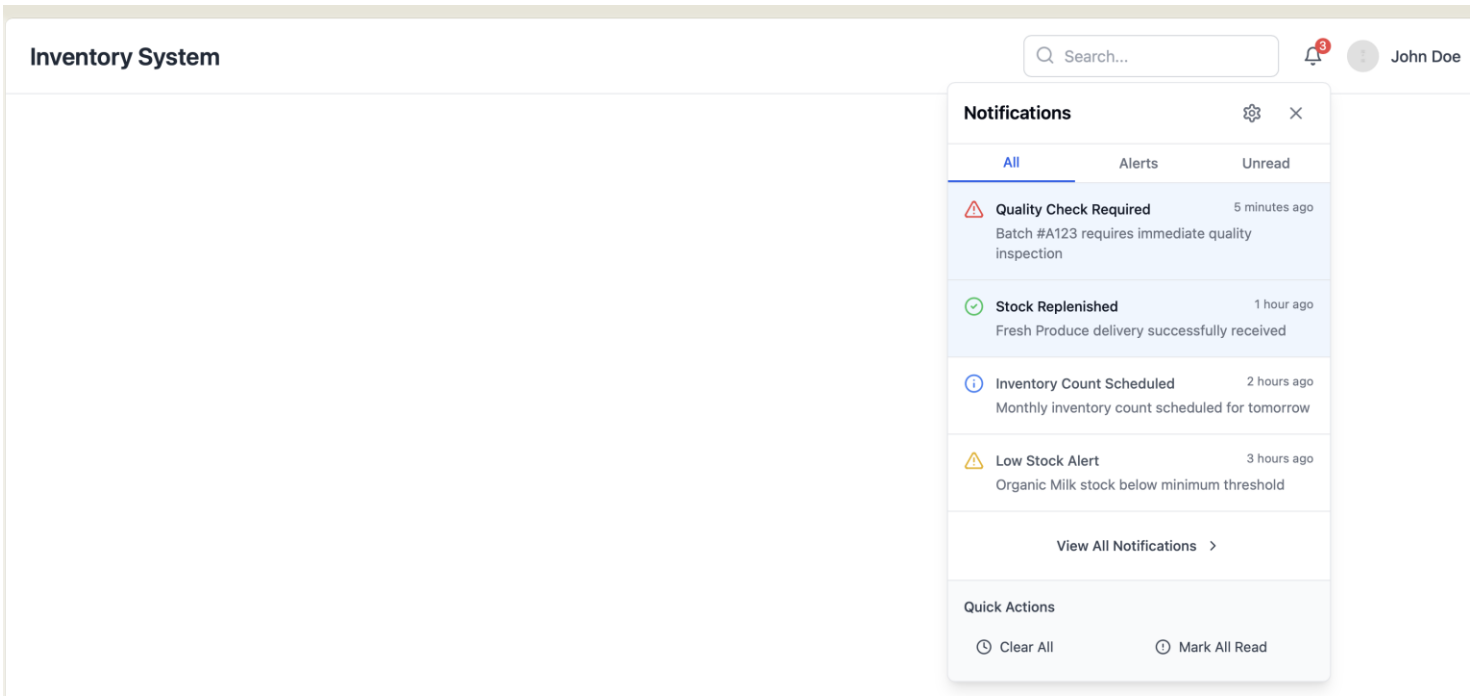
Forecast 3



Expiration Tracking



Stock Management



Notifications

3.5.6 Features of System Interface Modules

Main Dashboard

On the Main Page:

- Stock levels overview
- Quality alerts section
- Expiration warnings
- Forecast insights
- Menu selection on the left side of the dashboard
- Notifications on the toğ right of the dashboard

Stock Management

- Stock level details
- Product categorization
- Add new items button
- Filtering option

Quality Control

- Total number of inspections
- Failed number of inspections
- Pass rate
- Average response time of the inspection
- Defect detection results
- Deterioration tracking
- Alert management
- Filtering option
- Adding new item to be inspected option
- Exporting feature

Expiration Tracking

- Total expiring items
- Weekly expiring items
- Critical expiring items
- Filtering option

Forecasting Module

- Total projected sale
- Total stock level
- Estimated turn-over
- Stock-out risked items
- Sales forecast graph
- Inventory projection graph

Reports

- Data visualization
- Export feature
- Future forecast with time-series analysis of sales data

Notifications

- Quality check notification
- Critical expiring items notifications
- Inventory notifications
- Low stock alerts
- Marked as read and clear features

4. Glossary

- **CNN**: Convolutional Neural Network
- **OCR**: Optical Character Recognition
- **ML**: Machine Learning
- **MB**: Megabyte: a size unit
- **RTO**: System Recovery Time
- **API**: Application Programming Interface
- **SLA**: Service Level Agreement
- **GDPR**: General Data Protection Regulation
- **UI**: User Interface
- **PostgreSQL**: Open-source relational database
- **NoSQL**: Non-relational database
- **React**: JavaScript library for building user interfaces
- **PyTorch**: Machine learning framework
- **Python**: Programming language
- **Class Diagram:**
  - Represents the structure of a system.
  - Shows classes, their attributes, methods, and the relationships (e.g., inheritance, association) between classes.
  - Example: A Person class might have attributes like name and age and methods like speak() or walk().
- **Sequence Diagram:**
  - Represents the interaction between objects in a specific sequence over time.
  - Shows objects and the messages exchanged to complete a process or use case.
  - Example: A login process showing how a User sends credentials to AuthService, which verifies them and responds.
- **Use Case Diagram:**
  - Represents the functionalities of a system.
  - Shows actors (users or external systems) and their interactions with use cases (system functionalities).
  - Example: An online shopping system might have use cases like Browse Products, Add to Cart, and Checkout, with Customer as an actor.

5. References

- Project Proposal Document (CMPE491_Project_Proposal.pdf)
- Project Specification Report-2.pdf
- IEEE Code of Ethics (https://www.ieee.org/about/corporate/governance/p7-8.html)
- ACM Code of Ethics (https://www.acm.org/code-of-ethics)