LAMINA
STUDIOS

Data Science Internship

Individual Weekly Task Documentation

Week #5

Jimalyn B. Del Rosario

Table Contents

I. My Summarized Log

Encompasses five weekdays from Monday to Friday with 8-hours spent per day

DAY 21
Week 5
Monday
08/11/2025
[08:00 AM - 05:00 PM]

- Reviewed Smart Warehousing system design and assigned deliverables.
- Set sample data (inventory_alerts, rfid_logs, sales) using sample schema.
- Ran initial data ingestion.

DAY 22
Week 5
Tuesday
08/12/2025
[08:00 AM - 05:00 PM]

- Created dashboards:
  - Inventory Stock Trends (line chart)
  - RFID Scan Heatmap (by warehouse zone)
  - Table: Restock Triggered by Sensor Alerts
- Enabled auto-refresh on dashboards and verified live data through mock data generation.
- Integrated Flask API endpoint /sensor-alert to serve real-time inventory data.
- Began frontend setup for custom Chart.js dashboard with Next.js.

DAY 23
Week 5
Wednesday
08/13/2025
[08:00 AM - 05:00 PM]

- Implemented Chart.js component in InventoryChart.tsx using conditional formatting.
- Built dashboard.tsx page in Next.js to display Flask API data dynamically.
- Connected frontend to http://localhost:5001/sensor-alert using fetch.
- Tested dashboard display using mock and live data.
- Coordinated with AI and Web Dev teams for API integration and model updates.

DAY 24
Week 5
Thursday
08/14/2025
[08:00 AM - 05:00 PM]

- Polished frontend dashboard and ensured responsive layout using TailwindCSS.
- Enhanced Flask route logic for more structured JSON response.
- Verified full stack sync.
- Updated shared technical documentation with API summary, dev setup, and RFID diagrams.
- Tested Chart.js implementations for dashboard versatility.

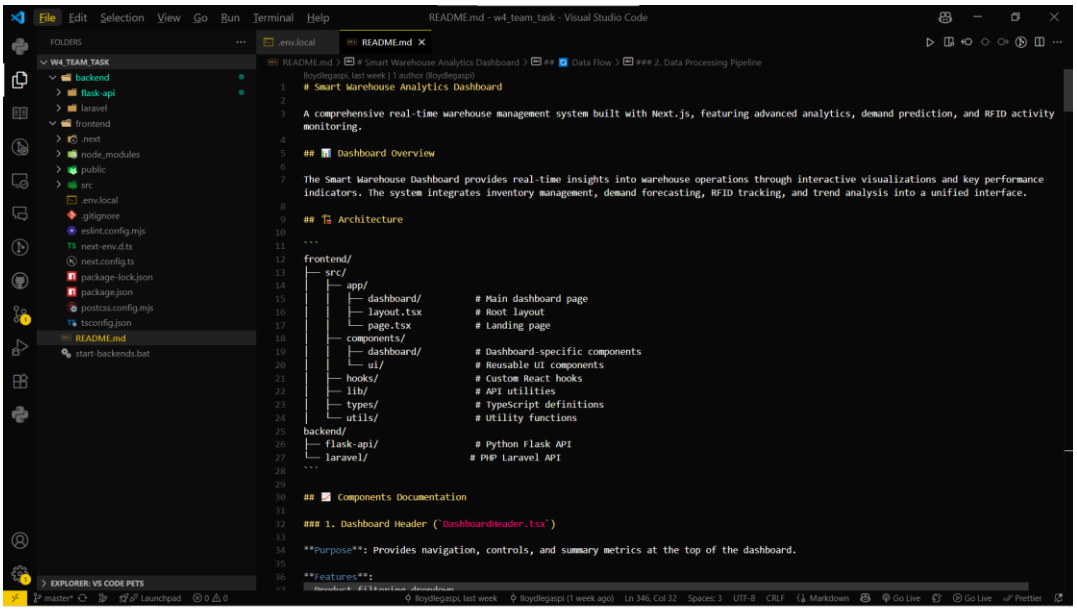
DAY 25
Week 5
Friday
08/15/2025
[08:00 AM - 05:00 PM]

- Finalized dashboards and embedded them into the warehouse insights interface.
- Validated accuracy of sensor-based stock alerts and forecasted demand charts.
- Documented Flask-to-Next.js data flow in shared documentation.
- Performed final UI/UX testing and pushed frontend code to GitHub.
- Submitted Week 5 deliverables: analytics dashboards and documentation updates.

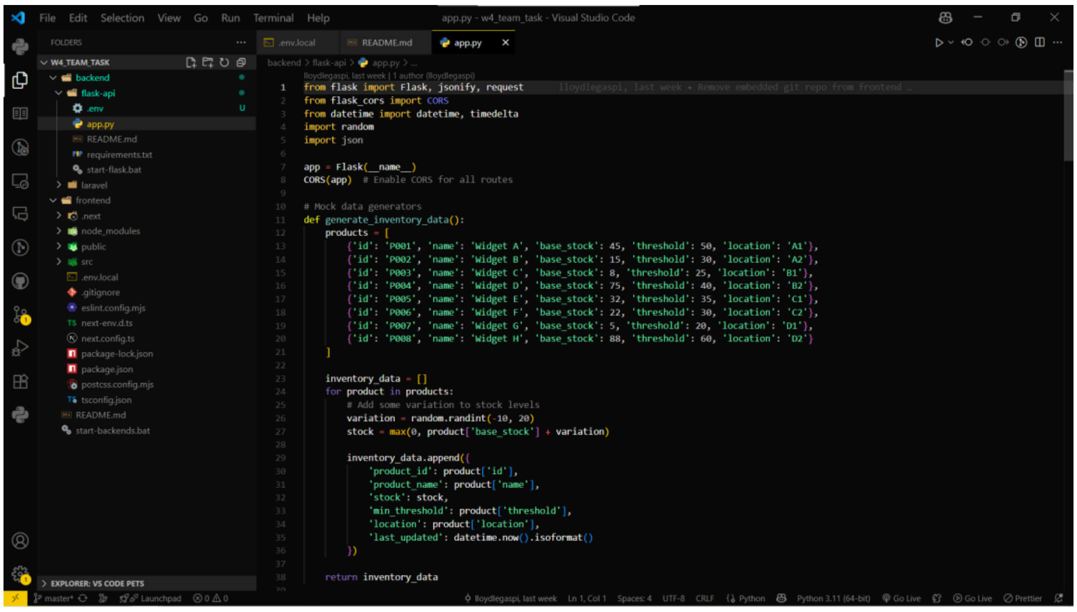
II. Team Task Progress Report

This is for Week 4: Smart Warehousing & Inventory Management

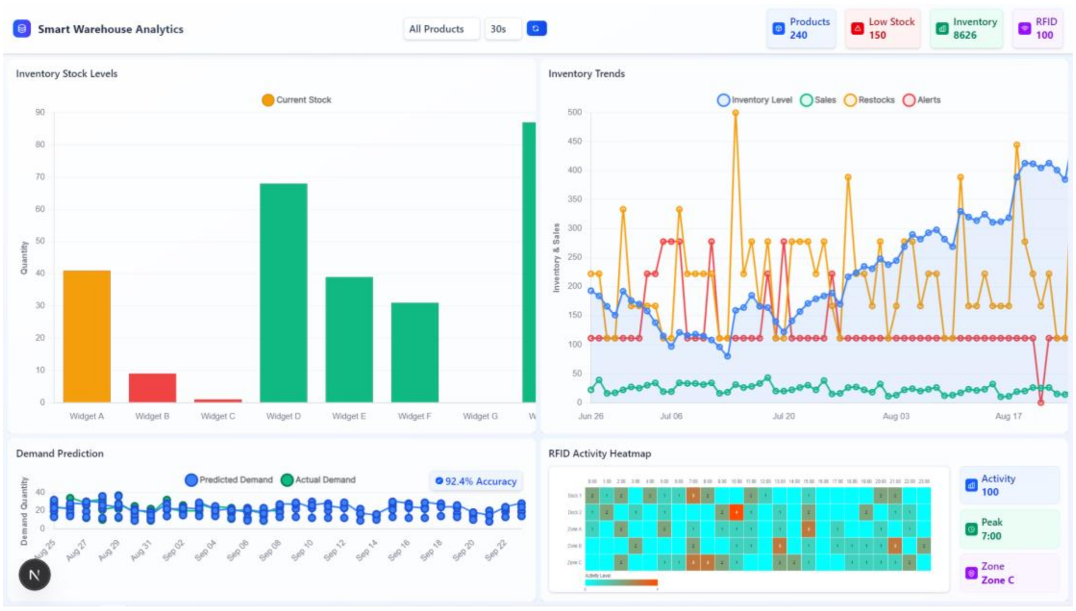
This task is done by team.



```
# Smart Warehouse Analytics Dashboard

A comprehensive real-time warehouse management system built with Next.js, featuring advanced analytics, demand prediction, and RFID activity
monitoring.

## 📊 Dashboard Overview

The Smart Warehouse Dashboard provides real-time insights into warehouse operations through interactive visualizations and key performance
indicators. The system integrates inventory management, demand forecasting, RFID tracking, and trend analysis into a unified interface.

## 🏗 Architecture

```
frontend/
├── src/
│   ├── app/
│   │   ├── dashboard/          # Main dashboard page
│   │   ├── layout.tsx          # Root layout
│   │   └── page.tsx            # Landing page
│   ├── components/
│   │   ├── dashboard/          # Dashboard-specific components
│   │   └── ui/                 # Reusable UI components
│   ├── hooks/                  # Custom React hooks
│   ├── lib/                    # API utilities
│   ├── types/                  # TypeScript definitions
│   └── utils/                  # Utility functions
backend/
├── flask-api/                  # Python Flask API
└── laravel/                  # PHP Laravel API
```

📈 Components Documentation

1. Dashboard Header (`DashboardHeader.tsx`)

Purpose: Provides navigation, controls, and summary metrics at the top of the dashboard.

Features:
```



```python
from flask import Flask, jsonify, request
from flask_cors import CORS
from datetime import datetime, timedelta
import random
import json

app = Flask(__name__)
CORS(app)  # Enable CORS for all routes

# Mock data generators
def generate_inventory_data():
    products = [
        {'id': 'P001', 'name': 'Widget A', 'base_stock': 45, 'threshold': 50, 'location': 'A1'},
        {'id': 'P002', 'name': 'Widget B', 'base_stock': 15, 'threshold': 30, 'location': 'A2'},
        {'id': 'P003', 'name': 'Widget C', 'base_stock': 8, 'threshold': 25, 'location': 'B1'},
        {'id': 'P004', 'name': 'Widget D', 'base_stock': 75, 'threshold': 40, 'location': 'B2'},
        {'id': 'P005', 'name': 'Widget E', 'base_stock': 32, 'threshold': 35, 'location': 'C1'},
        {'id': 'P006', 'name': 'Widget F', 'base_stock': 22, 'threshold': 30, 'location': 'C2'},
        {'id': 'P007', 'name': 'Widget G', 'base_stock': 5, 'threshold': 20, 'location': 'D1'},
        {'id': 'P008', 'name': 'Widget H', 'base_stock': 88, 'threshold': 60, 'location': 'D2'}
    ]

    inventory_data = []
    for product in products:
        # Add some variation to stock levels
        variation = random.randint(-10, 20)
        stock = max(0, product['base_stock'] + variation)

        inventory_data.append({
            'product_id': product['id'],
            'product_name': product['name'],
            'stock': stock,
            'min_threshold': product['threshold'],
            'location': product['location'],
            'last_updated': datetime.now().isoformat()
        })

    return inventory_data
```



Products 240
Low Stock 150
Inventory 8626
RFID 100

Activity 100
Peak 7:00
Zone Zone C