DIMA WMS – Technical Test Submission

This repository contains my solution for **Remote Software Engineer: Technical Test 01 – Inventory Management**.

The project consists of **two microservices**:

- **MSInventory** → Django REST Framework + PostgreSQL backend
- **MSWebclient** → Vue.js + TypeScript frontend

Both services implement inventory features for **DIMA WMS**, including CRUD operations, stock moves, multi-product support, and real-time inventory tracking.

Prerequisites

Ensure the following are installed:

- Python 3.10+
- Node.js v18+ & npm
- PostgreSQL 12.0+
- Git

■ Backend – MSInventory

The backend is built with **Django REST Framework**, handles inventory operations, and exposes **JWT-protected API endpoints**.

Swagger documentation is available for testing all endpoints.

1. Environment Variables

Create a .env file in msinventory/:

DEBUG=True
SECRET_KEY=QWESWESDSD1223EQDEF
DB_NAME=dima
DB_USER=postgres
DB_PASSWORD=admin
DB_HOST=127.0.0.1
DB_PORT=5432

2. Database Setup

```
# Create database
createdb -U postgres dima
# Or using psql:
psql -U postgres -c "CREATE DATABASE dima;"
# Optional: restore dump if provided
psql -U postgres -d dima < dump/dima.sql</pre>
```

3. Backend Setup & Commands

```
# Navigate to backend
cd msinventory
# Create virtual environment
python -m venv venv
# Activate virtual environment
# Windows:
venv\Scripts\activate
# macOS/Linux:
source venv/bin/activate
# Install dependencies
pip install -r requirements.txt
# Apply migrations
python manage.py migrate
# Create superuser (if not auto-created)
python manage.py createsuperuser
# Start development server
python manage.py runserver
```

Backend available at http://localhost:8000/

Swagger API docs: http://localhost:8000/swagger/

Admin panel: http://localhost:8000/admin/

Default Admin Credentials:

Username: admin
Password: admin123

4. Backend Features

- ✓ Products CRUD
- ✓ Locations CRUD
- ✓ Stock move with multiple products (optional task)
- ✓ Real-time Inventory Levels per product/location

5. Backend Project Structure

```
msinventory/
                           # All Django apps
  - inventory/
      -- locations/
       - products/
     -- snapshots/
       - stockmoves/
    L__ suppliers/
  - msinventory/
                           # Main Django project
  - utils/
                           # Utility functions (exceptions, helpers, JWT)
  - env/
 — en.,
— flakes/
  -- app.yaml

    load initial data.py

  manage.py
  - README.md
  requirements.txt
```

(Each app under inventory/ contains migrations, models, serializers, views, urls, and tests as needed)

⊕ Frontend – MSWebclient

The frontend is built with **Vue.js** + **TypeScript** and communicates with the backend via API.

1. Environment Variables

```
Create a .env file in mswebclient/:
```

```
VITE_API_BASE_URL=http://localhost:8000/api
VITE APP VERSION=1.20.3
```

2. Frontend Setup & Commands

```
# Navigate to frontend
cd mswebclient

# Install dependencies
npm install

# Start development server
npm run dev

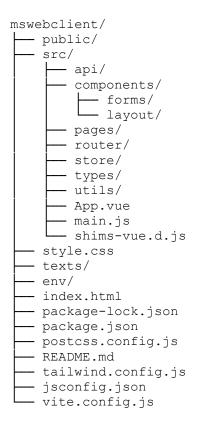
# For production build
npm run build
```

Frontend available at http://localhost:8080/

3. Frontend Features

- ✓ Real-time Inventory Level display
- ✓ Exception handling & user-friendly messages

4. Frontend Project Structure



(src/components/forms contains form components; src/pages contains page components for CRUD operations)

Authentication

- JWT-protected backend endpoints
- Obtain token via:

"username": "admin",
"password": "admin123"

Additional Useful Commands

```
# Backend tests
python manage.py test

# Create new migrations after model changes
python manage.py makemigrations

# Load initial data (if implemented)
python manage.py loaddata initial_data.json

# Frontend tests
npm run test:unit

# Frontend linting
npm run lint
```

Access Points

- Backend API: http://localhost:8000
 Frontend App: http://localhost:8080
- Admin Panel: http://localhost:8000/admin
- API Documentation (Swagger): http://localhost:8000/swagger

The backend must be running before the frontend can make successful API calls. Both servers can run simultaneously on different ports.