## Warehouse Management System

## Problem Requirements

### 1. ****Project Overview****

The **SmartWarehouse Management System (WMS)** is designed to efficiently handle product inventory and stock movement across multiple warehouse locations. The system enables administrators and employees to manage products, track stock changes, and maintain warehouse information using role-based access control.

### 2. ****Problem Scenario****

Currently, stock operations are tracked manually in spreadsheets, leading to:

* Inventory mismatches
* Missing product updates
* Lack of visibility across warehouses
* Unrecorded inbound and outbound stock movements

The SmartWarehouse WMS solves these challenges by providing:

* Centralized inventory control
* Automated stock change logging
* Role-based access for secure operations
* RESTful APIs for product, stock, and warehouse management

### 3. ****Functional Requirements (APIs & Use Cases)****

#### **Users**

* **POST /registerUser** — Register a new system user and assign a role (ADMIN, MANAGER, EMPLOYEE).
* **POST /login** — Authenticate a user and start a session/issue a token.
* **GET /getAllUsers** — Retrieve a list of all users.

#### **Warehouses**

* **POST /createWarehouse** — Add a new warehouse with its address details.
* **GET /getWarehouseById** — Retrieve specific warehouse details by ID.
* **GET /getAllWarehouses** — View all warehouses in the system.
* **PUT /updateWarehouse** — Modify warehouse information.
* **DELETE /deleteWarehouse** — Remove a warehouse record.

#### **Products**

* **POST /createProduct** — Add a new product with category and price.
* **GET /getProductById** — Retrieve a single product’s details by ID.
* **GET /getAllProducts** — List all available products.
* **PUT /updateProduct** — Edit product details.
* **DELETE /deleteProduct** — Remove a product.
* **GET /getProductsByCategory** — List products filtered by category.
* **GET /getProductsByPriceRange** — Retrieve products within a specific price range.

#### **Stocks**

* **POST /restock** — Add quantity to stock and log it as an **INBOUND** change.
* **GET /getTotalStockByProductId** — Check total stock quantity for a product.

#### **Orders**

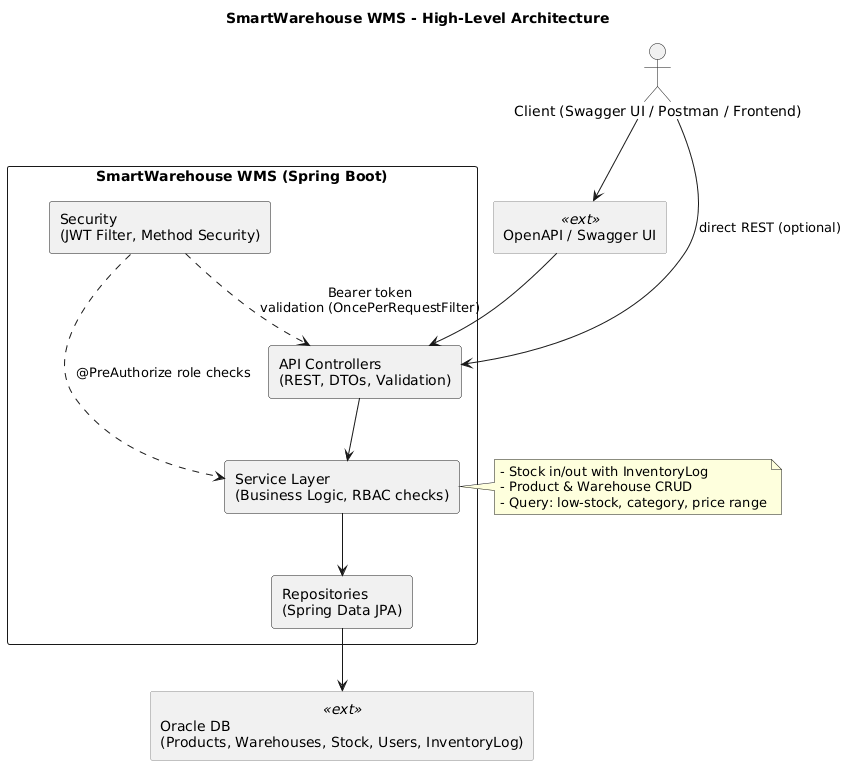
* **POST /sendOrderToWarehouse** — Deduct product quantity from warehouse (**OUTBOUND**).
* **POST /returnOrderToWarehouse** — Add returned items back to warehouse stock (**INBOUND**).

### 4. ****Non-Functional Requirements****

* **Performance:** Real-time stock updates (under 2 seconds).
* **Security:** JWT-based authentication and role-based authorization.
* **Scalability:** Multiple warehouses with independent stock tracking.
* **Availability:** REST API accessible 24/7.
* **Logging:** All stock activities are recorded in InventoryLog.

### 5. ****Class Diagram****

### 6. ****High-level Diagram****



### 7. ERD