**External Design Document**

**Project Title: Warehouse stock management**

**Members:**

**Minh Doan Nguyen**

**Rona Emre Altug**

**Erdem Vural**

**Louis | Lu-Hsiang Wang**

**Jaspreet Singh**

-------------------------------------------------------------------------------------

**Table of Contents**

1. **Overview**
2. **System Architecture**
3. **Frontend (React + Redux Toolkit)**
4. **Backend (Node.js + Express)**
5. **Database (MongoDB)**
6. **API Routes**
7. **Environment Configuration**
8. **Setup & Deployment**

-------------------------------------------------------------------------------------

**1. Overview**

This application is a **Warehouse stock management** designed with a modular architecture. It includes:

* A **React**-based frontend.
* A **Node.js + Express** backend.
* A **MongoDB** database.
* Full **authentication** and **authorization** system.
* Dockerized setup for simplified deployment.

This application is focus to help managing a warehouse, such as checking is a product still exist in it, or how many it has, and adding or deleting product.

-------------------------------------------------------------------------------------

图示

AI 生成的内容可能不正确。**2. System Architecture**

**High-Level Diagram**

Client

|

Server

|

MongoDB

**Deployment**

* Docker-based containerized deployment.
* Services: frontend (port 3000), backend API (port 5000), MongoDB (27017).

-------------------------------------------------------------------------------------

**3. Frontend (React + Redux Toolkit)**

**Location:** app/src/

**Key Features**

* Redux for global state management.
* RTK Query for API handling.
* Auth management with JWT.
* Product list, creation, and update flows.

**Notable Files**

| **File** | **Purpose** |
| --- | --- |
| store.js | Redux store setup |
| apiConfig.js | API base URLs |
| authSlice.js | Authentication state logic |
| productsApiSlice.js | Product endpoints and queries |
| useAuth.js | Hook to handle auth context |
| auth.js | JWT token utilities |

-------------------------------------------------------------------------------------

**4. Backend (Node.js + Express)**

**Location:** server/src/

**Key Features**

* REST API endpoints for:
  + Authentication
  + Product management
  + User management
* JWT-based authentication & refresh token system.

**Notable Files**

| **File** | **Purpose** |
| --- | --- |
| server.js | App entry point |
| db.js | MongoDB connection logic |
| passport.js | Passport setup for auth |
| authController.js | Handles login, register, token refresh |
| productController.js | Product CRUD logic |
| userController.js | User-related operations |
| authMiddleware.js | Protects private routes |
| uploadMiddleware.js | Handles image/file uploads |

-------------------------------------------------------------------------------------

**5. Database (MongoDB)**

**Models**

| **Model** | **File** | **Description** |
| --- | --- | --- |
| User | userModel.js | Email, password (hashed), role, etc. |
| Product | productModel.js | Name, description, price, image |
| RefreshToken | refreshTokenModel.js | Stores refresh tokens for JWT renewal |

-------------------------------------------------------------------------------------

**6. API Routes**

| **Route** | **Method** | **Description** |
| --- | --- | --- |
| /api/auth/register | POST | Register new user |
| /api/auth/login | POST | Login user |
| /api/auth/refresh-token | POST | Renew access token |
| /api/products/ | GET/POST | List or create product |
| /api/products/:id | PUT/DELETE | Update or delete product |
| /api/users/ | GET | List users (admin only) |

-------------------------------------------------------------------------------------

**7. Environment Configuration**

.env contains keys like:

* PORT
* MONGO\_URI
* JWT\_SECRET
* REFRESH\_TOKEN\_SECRET

-------------------------------------------------------------------------------------

**8. Setup & Deployment**

**Dockerized Setup**

docker-compose up -d

**Access**

* Frontend: http://localhost:3000
* Backend: http://localhost:5000
* MongoDB: mongodb://localhost:27017