

Inventory Management Project

Project Overview

This is a **full-stack Inventory Management application** built with **ASP.NET Core Web API** for the backend and **React** for the frontend. It allows users to manage products, categories, and perform CRUD operations with a simple and user-friendly interface.

The project demonstrates the integration of a modern frontend with a robust backend API.

Technologies Used

Backend

- **Framework:** ASP.NET Core 8.0
- **Packages:**
 - Newtonsoft.Json for JSON handling
 - Swashbuckle.AspNetCore for Swagger API documentation
- **Data Storage:** JSON file (Data/data.json) for simplicity
- **Features:**
 - RESTful API endpoints for products and categories
 - Swagger integration for API testing and documentation

Frontend

- **Framework:** React 19
- **UI & Styling:**
 - @mui/material (Material UI components)
 - @mui/icons-material (Material Icons)
 - Bootstrap 5 for styling
 - TailwindCSS for utility-first styling
- **Form Validation:** Yup
- **HTTP Requests:** Axios
- **Notifications:** react-toastify

- **Routing:** react-router-dom
 - **Testing Libraries:**
 - @testing-library/react
 - @testing-library/jest-dom
 - @testing-library/user-event
 - @testing-library/dom
-

Project Structure

Backend (ASP.NET Core)

/InventoryAPI

/Controllers

- ProductsController.cs
- CategoriesController.cs

/Data

- data.json

/Models

- Product.cs
- Category.cs

/Services

- DataService.cs

Program.cs

Startup.cs

Frontend (React)

```
/Frontend
  /src
    /assets
      -Inventory.png
    /components
      - CategoryDialog.jsx
      -Navbar.jsx
      - ProductDialog.jsx
    /pages
      - Dashboard.jsx
      - Products.jsx
      - Categories.jsx
      - ActiveProducts.jsx
      -LowStockProducts.jsx
      - CategoryProducts.jsx
    api.js
    App.js
    index.js
  package.json
```

Installation & Setup

Backend

1. Navigate to the backend folder: **cd backend/InventoryAPI/ InventoryAPI**
2. Restore dependencies: **dotnet restore**
3. Run the backend API: **dotnet run**
4. API will be available at **https://localhost:7128**

5. Swagger documentation available at **https://localhost:7128/swagger**

Frontend

1. Navigate to the frontend folder: **cd frontend**
 2. Installation dependencies: **npm install**
 3. Start the development server: **npm start**
 4. Open your browser at **http://localhost:3000**
-

Features

- Manage products and categories
 - Add, edit, delete products and categories
 - Real-time form validation with **Yup**
 - User notifications with **react-toastify**
 - Clean UI using **Material UI**, and **Bootstrap**.
 - API documentation via **Swagger**
 - In this section, the user can select products based on categories.
-

Future Improvements

- Replace JSON file with a proper database (SQL Server or MongoDB)
 - Add authentication and authorization
 - Add advanced search, filter, and pagination
 - Implement role-based access control
-

Notes

- Ensure backend API is running before using the frontend
 - Data is currently stored in Data/data. Json for simplicity
 - API base URL is configured in frontend/src/api/index.js
-