# **Web-Based Product Display Application**

## **Project Overview**

This document outlines a web-based platform that enables users to view and interact with product information. The application is designed for both regular users and administrators (managers/super users).

## **Features**

### **User Interface (UI)**

* **Product Listing Page:**
  + Navigation bar displaying user information.
  + List of products featuring images, descriptions, categories, and prices (in South African Rand - R).
  + “Add to Cart” functionality with toast notifications upon successful addition.
  + Filtering options by category (e.g., fruits, vegetables).
  + Search bar for quick product name searches.
  + Cart section displaying the number of items and total price (dynamically updated).
* **Detailed Product View (Super User/Manager Only):**
  + Accessible by clicking on a product in the listing page.
  + Sales Information section including:
    - A line chart representing sales quantity over time.
    - A sales table summarizing sale price, date, and quantity sold.
    - Total sales and total quantity sold.
* **User Profile Section:**
  + Displays user information (name, role, email).
  + Option to update user profile details.
* **Logout Functionality:**
  + Logout button clears user data from local storage and redirects to the home page.

## **Authentication**

* Secure login using JWT (JSON Web Token) authentication.
* User data is fetched from local storage, including role and email.

## **Technologies**

* **Frontend:** React.js
* **Backend:** ASP.NET Core API
* **Database:** SQL Server
* **API Endpoints:**
  + **Products:** Get product information and sales data.
  + **Accounts:** User login, registration, and email confirmation.
* **Additional Enhancements:**
  + Navigation using react-router-dom for dynamic navigation to specific product sales history.
  + Secure email server through Mailhog.
  + User-friendly error messages for validation.
  + React for building UI components.
  + CSS Modules for component styling.
  + Axios for making API calls to fetch data.
  + Material UI Icons for visual enhancements.

## **How the App Runs**

### **1. Package Installation**

# Navigate to the project directory and run the following command:  
npm install

### **2. Setup**

1. Open two separate terminal windows.
2. In the first terminal, navigate to the API directory and execute:

* dotnet run

1. In the second terminal, run React using:

* npm start

1. Start Mailhog for handling email confirmations.

### **3. Using the App**

* **Registration:** Register as a user. You will receive an email in Mailhog for account confirmation.
* **Home Page:** Log in after confirmation to access the home page for product browsing.
* **Detailed Product Page:** Log in as a manager/admin to view detailed sales information with graphs.

## **Admin Credentials (for full app experience)**

* **Email:** admin@example.com
* **Password:** AdminPassword@1

## **Additional Notes**

This document serves as the foundational guide for the application. Specific implementation details may vary based on development choices.