

PRODUCT INVENTORY MANAGEMENT

IIHT

Time To Complete: 10 to 12 hr

CONTENTS

1	Project Abstract	3
2	Problem Statement	3
3	Proposed Product Inventory Management System Wireframe	4
3.1	Welcome Page	4
3.2	Screenshots	5
4	Business-Requirement:	15
5	Constraints	21
6	Mandatory Assessment Guidelines	22

1 PROJECT ABSTRACT

In the ever-evolving field of retail and business management, there is a growing necessity for digital platforms that facilitate efficient inventory tracking, enhance product management capabilities, and empower businesses to optimize their operations. With this vision in mind, the CEO of a product management startup assigns a team of developers with the task of creating a **Product Inventory Management System** using Angular.

Your task is to develop a digital solution that allows businesses to manage their product inventory efficiently by adding, updating, deleting, and listing products. The system should enable users to organize products by categories, update stock quantities, and manage product details such as descriptions, prices, and manufacturers. Additionally, the platform should allow administrators to securely log in, perform inventory actions, and track their inventory in real time.

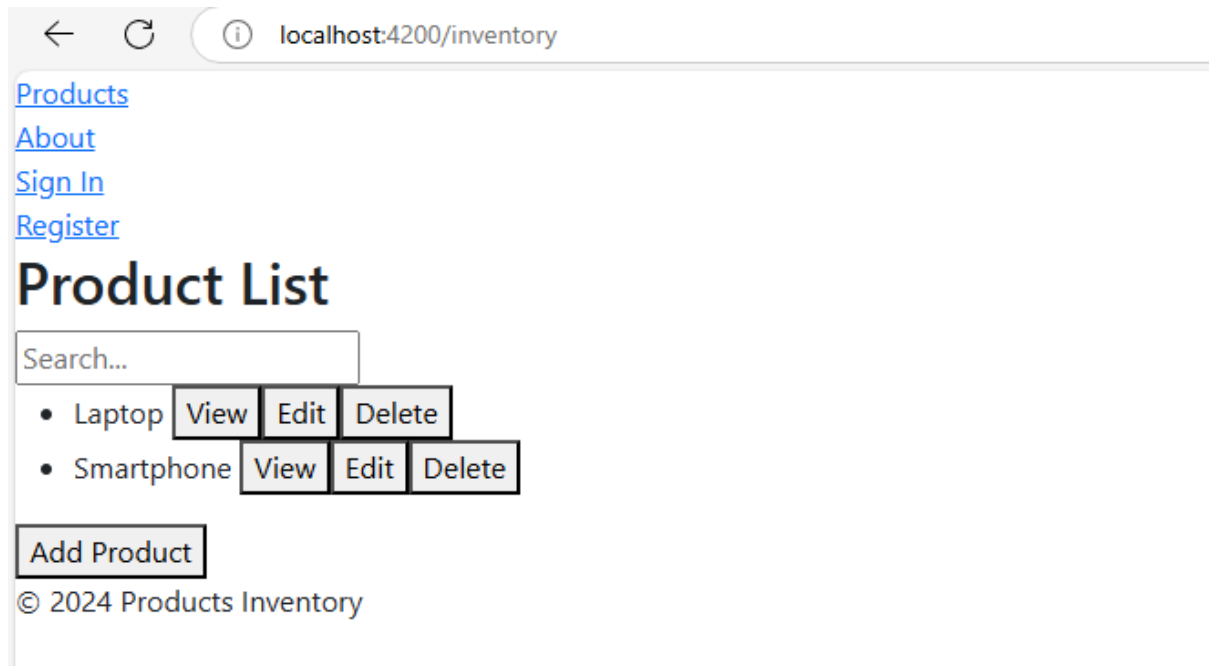
2 PROBLEM STATEMENT

“Product Inventory Management System” is a Single Page Application (SPA) designed to enhance inventory management for businesses by simplifying the process of managing products and stock. This system enables businesses to efficiently add, update, delete, list, and search for product records. Users can also categorize products, update stock quantities, and view detailed product information such as descriptions, prices, and manufacturers.

3 PROPOSED PRODUCT INVENTORY MANAGEMENT SYSTEM WIREFRAME

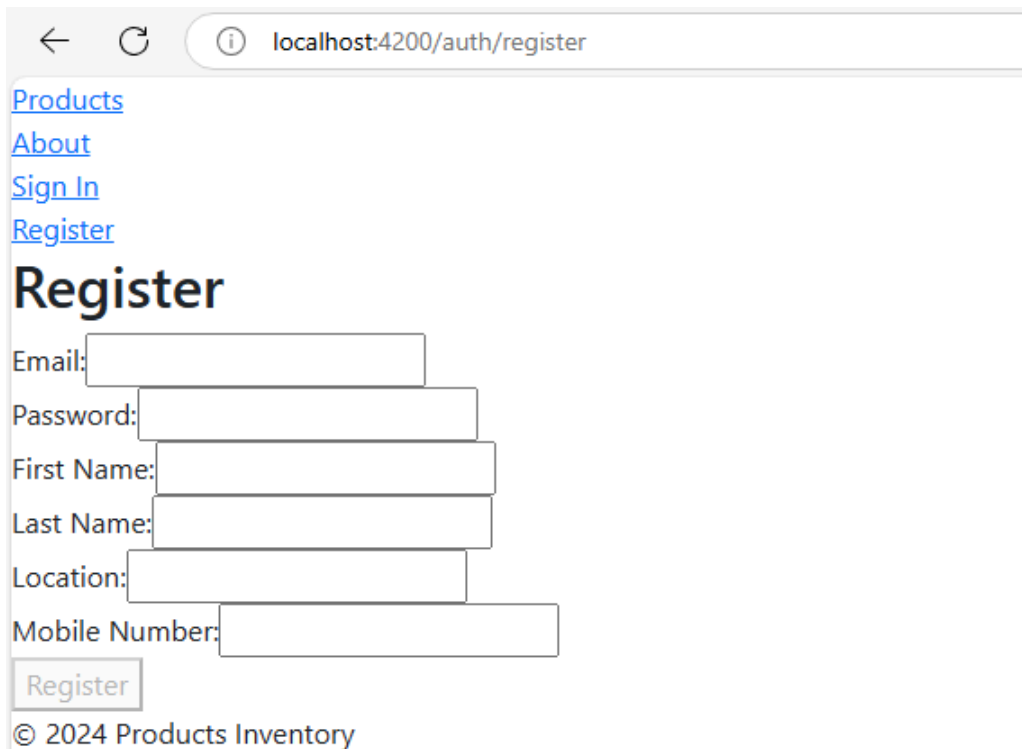
UI needs improvisation and modification as per given use case and to make test cases passed.

3.1 WELCOME PAGE

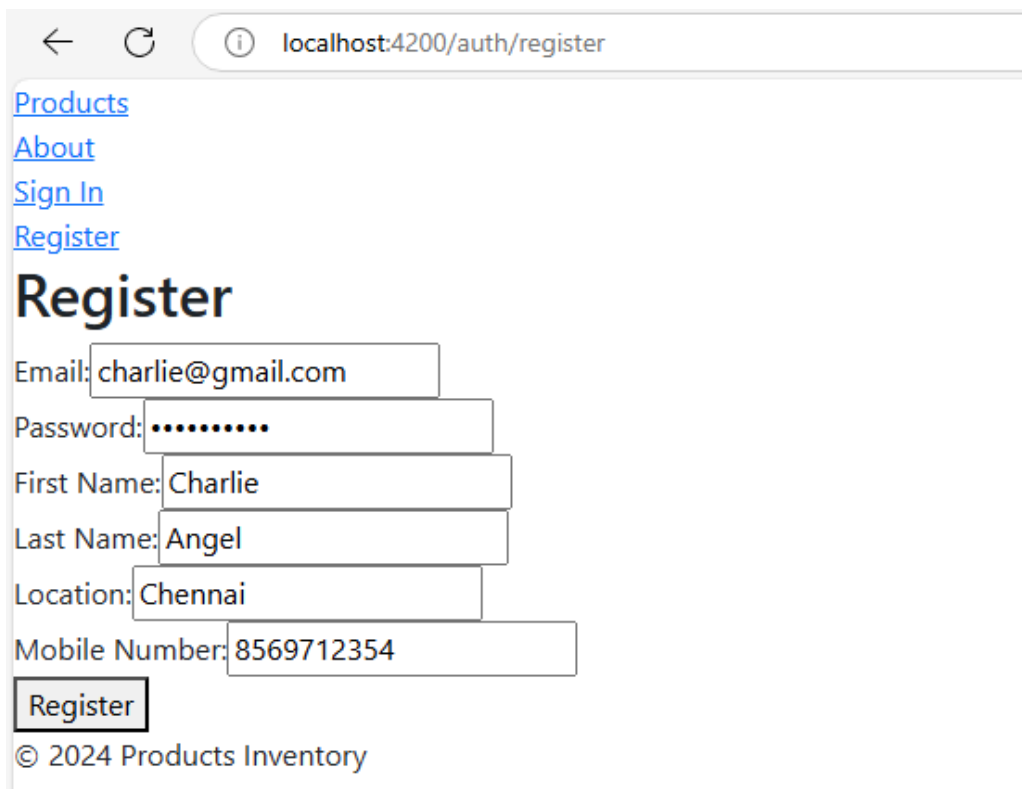


3.2 SCREENSHOTS

Register



A screenshot of a web browser showing the 'Register' page. The browser's address bar displays 'localhost:4200/auth/register'. On the left, there is a navigation menu with links: 'Products', 'About', 'Sign In', and 'Register'. The main heading is 'Register'. Below it, there are six empty input fields for 'Email:', 'Password:', 'First Name:', 'Last Name:', 'Location:', and 'Mobile Number:'. A 'Register' button is positioned below the fields. At the bottom, the footer text reads '© 2024 Products Inventory'.



A second screenshot of the same 'Register' page, but with the form fields filled out. The 'Email:' field contains 'charlie@gmail.com', 'Password:' contains eight dots, 'First Name:' contains 'Charlie', 'Last Name:' contains 'Angel', 'Location:' contains 'Chennai', and 'Mobile Number:' contains '8569712354'. The 'Register' button is now highlighted with a black border. The rest of the page, including the navigation menu and footer, remains the same.

← ↻ ⓘ localhost:4200/auth/register

[Products](#)
[About](#)
[Sign In](#)
[Register](#)

Register

Email:

Password:

First Name:

Last Name:

Location:

Mobile Number:

© 2024 Products Inventory

localhost:4200 says
Registration successfull!

Sign In

← ↻ ⓘ localhost:4200/auth/sign-in

[Products](#)
[About](#)
[Sign In](#)
[Register](#)

Sign In

Email:

Password:

© 2024 Products Inventory

*** Already added some users, kindly use their credentials in db.json file ***

← ↻ ⓘ localhost:4200/auth/sign-in

[Products](#)
[About](#)
[Sign In](#)
[Register](#)

Sign In

Email:

Password:

© 2024 Products Inventory

Signed In as a User

← ↻ ⓘ localhost:4200/inventory

[Products](#)
[About](#)

Product List

- Laptop
- Smartphone

© 2024 Products Inventory

Add Product

← ↻ ⓘ localhost:4200/inventory/add-product

[Products](#)
[About](#)

Logout

Add Product

Name:

Description:

Manufacturer:

Price:

Quantity:

Add Product

© 2024 Products Inventory

Already added some products with their details

← ↻ ⓘ localhost:4200/inventory/add-product

[Products](#)
[About](#)

Logout

Add Product

Name:

Description:

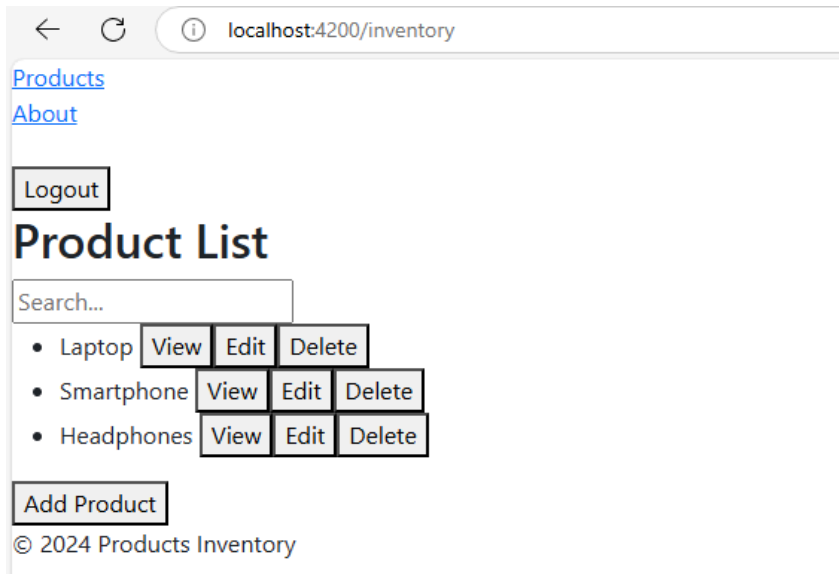
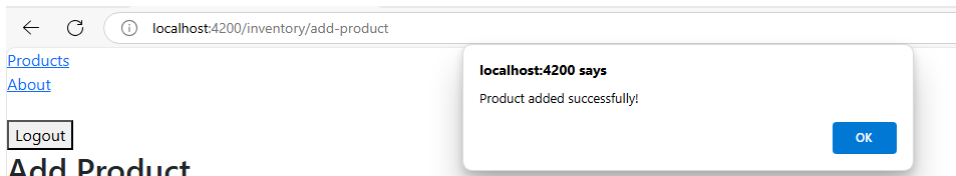
Manufacturer:

Price:

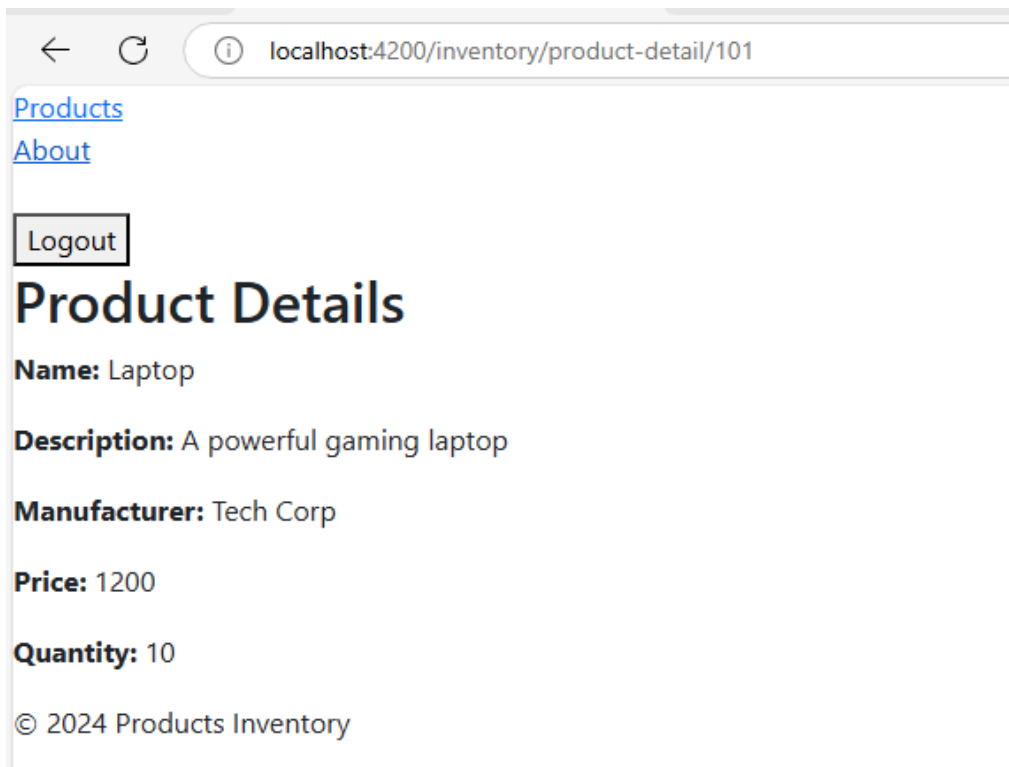
Quantity:

Add Product

© 2024 Products Inventory



*****View Product*****



Edit Product

← ↻ ⓘ localhost:4200/inventory/update-product/101

[Products](#)
[About](#)

Logout

Update Product

Name:

Description:

Manufacturer:

Price:

Quantity:

Update Product

© 2024 Products Inventory

← ↻ ⓘ localhost:4200/inventory/update-product/101

[Products](#)
[About](#)

Logout

Update Product

Name:

Description:

Manufacturer:

Price:

Quantity:

Update Product

© 2024 Products Inventory

← ↻ ⓘ localhost:4200/inventory/update-product/101

[Products](#)
[About](#)

Logout

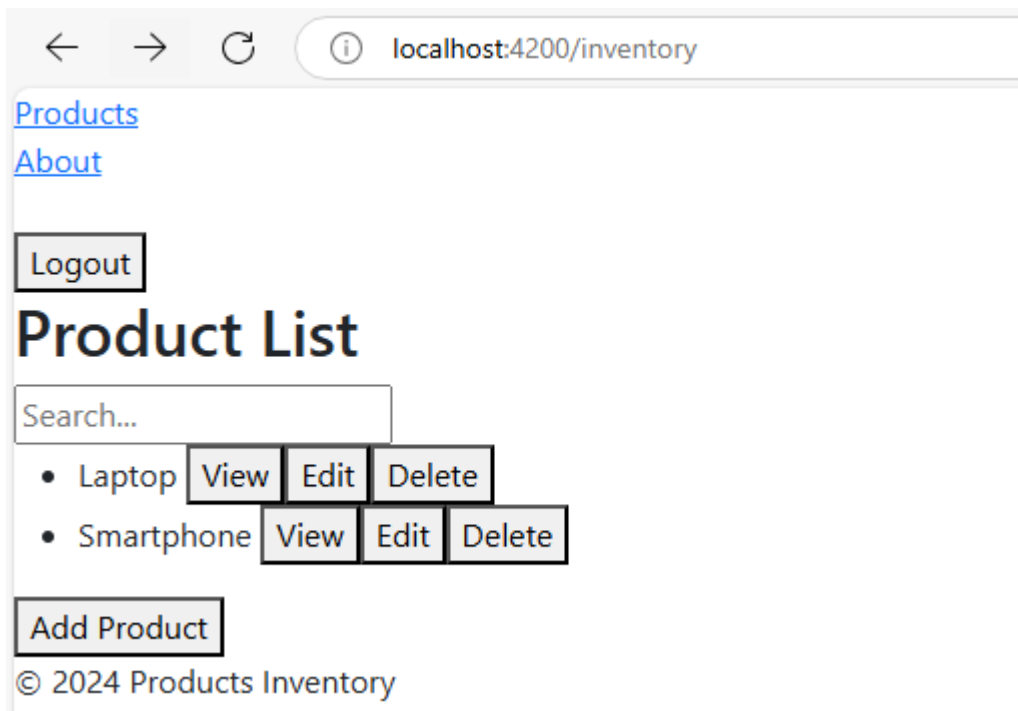
Update Product

Name:

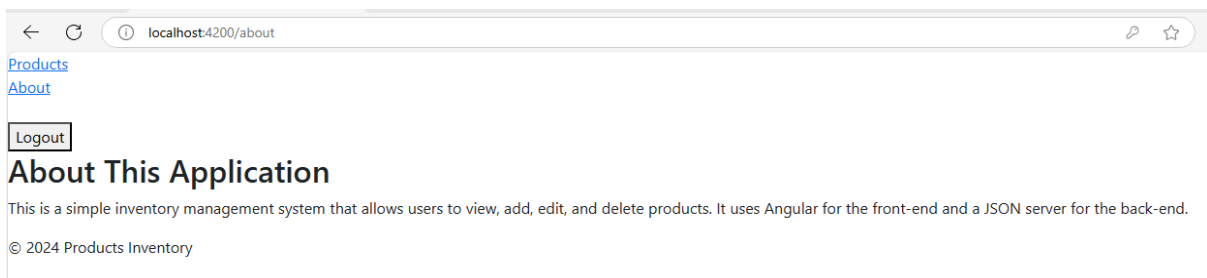
localhost:4200 says
Product updated successfully!

OK

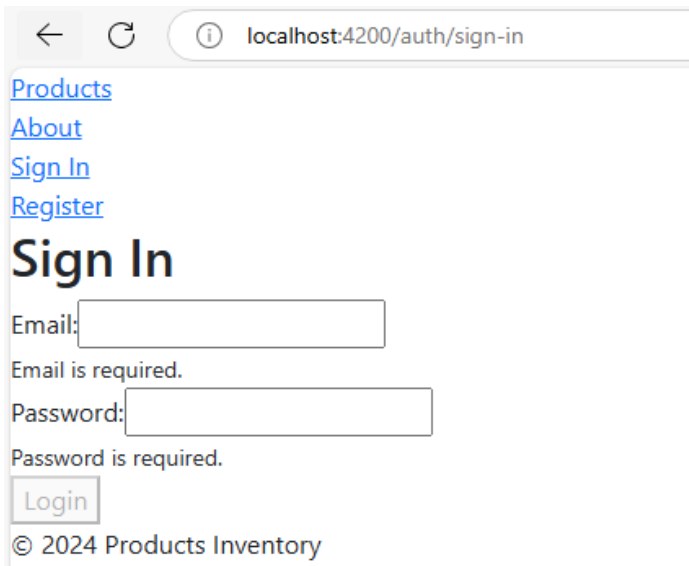
Delete Product



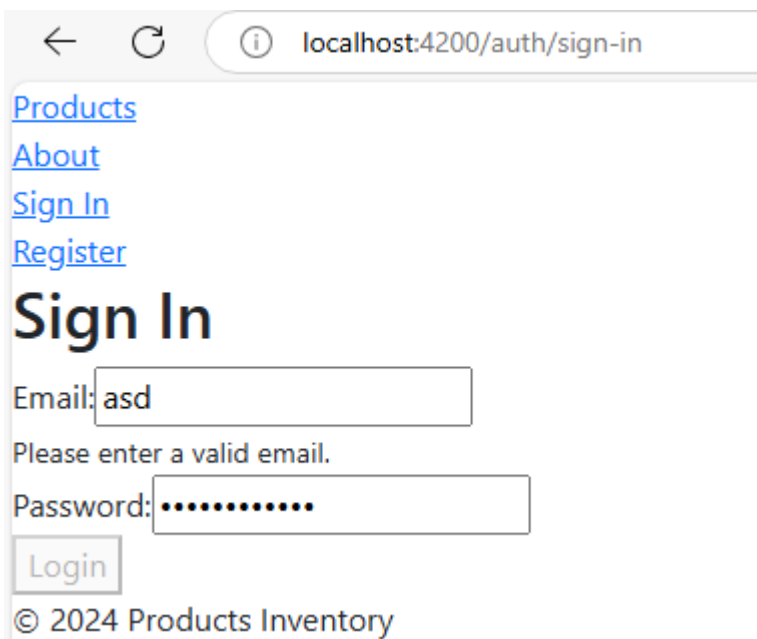
About



Validations



A screenshot of a web browser window at the URL `localhost:4200/auth/sign-in`. The page has a navigation menu with links for [Products](#), [About](#), [Sign In](#), and [Register](#). The main heading is "Sign In". Below it, there are two input fields: "Email:" and "Password:". The "Email:" field is empty, and below it is the message "Email is required.". The "Password:" field is also empty, and below it is the message "Password is required.". At the bottom of the form is a "Login" button. The footer text is "© 2024 Products Inventory".



A screenshot of the same web browser window after a sign-in attempt. The "Email:" field now contains the text "asd", and below it is the message "Please enter a valid email.". The "Password:" field contains ten dots, indicating a password has been entered. The "Login" button is still visible. The footer text remains "© 2024 Products Inventory".

← ↻ ⓘ localhost:4200/auth/register

[Products](#)
[About](#)
[Sign In](#)
[Register](#)

Register

Email:
Email is required.

Password:
Password is required.

First Name:
First name is required.

Last Name:
Last name is required.

Location:
Location is required.

Mobile Number:
Mobile number is required.

© 2024 Products Inventory

← ↻ ⓘ localhost:4200/inventory/add-product

[Products](#)
[About](#)

Add Product

Name:
Product name is required.

Description:
Description is required.

Manufacturer:
Manufacturer is required.

Price:
Price is required.

Quantity:
Quantity is required.

© 2024 Products Inventory

[←](#) [↻](#) [localhost:4200/auth/register](#)

[Products](#)
[About](#)
[Sign In](#)
[Register](#)

Register

Email:

Please enter a valid email.

Password:

Password must be at least 6 characters long.

First Name:

Last Name:

Location:

Mobile Number:

Enter a valid 10-digit mobile number.

© 2024 Products Inventory

4 BUSINESS-REQUIREMENT:

As an application developer, develop the Product Inventory Management System (Single Page App) with below guidelines:

User Story #	User Story Name	User Story
US_01	Welcome Page	<p>As a user I should be able to visit the welcome page as the default page.</p> <p>Acceptance criteria:</p> <ol style="list-style-type: none">There should be 4 routes/links:<ul style="list-style-type: none">Products: When clicking this link, it should navigate to the home page displaying the product list.About: When clicking this link, it should navigate to the "About" page, providing details about the application.Sign In: When clicking this link, it should navigate to the login page.Register: When clicking this link, it should navigate to the registration page. <p>The above-mentioned 4 navigation routes/links should be present and functional across all the respective pages of the Product Inventory System, allowing easy access to any section from anywhere within the app until logging in.</p> <ol style="list-style-type: none">The Welcome Page should display the title "Product List" as heading in h2.A search bar should be available to filter the products.A list of products should be displayed, where each product includes:<ul style="list-style-type: none">Product Name (e.g., Laptop, Smartphone).Action Buttons:<ol style="list-style-type: none">ViewEditDeleteAdd Product Button: Clicking this button should redirect to the Sign In page until logged in.Any action (View, Edit, Delete, or Add Product) should redirect the user to the Sign In page if they are not logged in.

		<p>7. You need to use the already implemented footer component to ensure that "© 2024 Products Inventory" is displayed on all pages, including after logging in.</p> <p>** Kindly refer to the screenshots for any clarifications. **</p>
	About Page	<p>Acceptance criteria:</p> <ol style="list-style-type: none"> 1. You need to use the already implemented About Page link, it should be able to view the details about the application on the "About" page including after logging in. <p>The Products, About, Sign In, and Register links should be present throughout all the respective pages in the application until logging in.</p> <p>** Kindly refer to the screenshots for any clarifications. **</p>
	Sign In	<p>Acceptance criteria:</p> <ol style="list-style-type: none"> 1. The page should display the title "Sign In" as heading in h2. 2. The page should display a Login form that includes the following fields: <ul style="list-style-type: none"> ● Email: Required field, should display a validation message if left empty as mentioned in screenshots above. ● Password: Required field, should display a validation message if left empty as mentioned in screenshots above. 3. The form should include a Login button to submit the login credentials: <ul style="list-style-type: none"> ● Should be disabled by default. ● Should be enabled only when both fields are filled correctly. 4. The system should validate the user credentials upon submission and provide appropriate feedback: <ul style="list-style-type: none"> ● If the credentials are wrong, it should display as mentioned in the screenshot. ● Upon successful login, the user should be redirected to the Home Page. <p>The Products, About, Sign In, and Register links should be present throughout all the respective pages in the application until logging in.</p>

	Register	<p>** Kindly refer to the screenshots for any clarifications. **</p> <p>Acceptance criteria:</p> <ol style="list-style-type: none"> 1. The Register Page should display the title: "Register" as heading in h2. 2. The form should include the following fields: <ul style="list-style-type: none"> ● Email: Required field, should display a validation message if left empty as mentioned in the screenshot. ● Password: Required field, should display a validation message if left empty as mentioned in the screenshot. ● First Name: Required field, should display a validation message if left empty as mentioned in the screenshot. ● Last Name: Required field, should display a validation message if left empty as mentioned in the screenshot. ● Location: Required field, should display a validation message if left empty as mentioned in the screenshot. ● Mobile Number: Required field, should display a validation message if left empty as mentioned in the screenshot. 3. The form should have an "Register" button: <ul style="list-style-type: none"> ● Should be disabled by default. ● Should be enabled only when all fields are filled correctly. 4. Upon successful registration, the user's details should be registered in db and allowed to sign in. <p>The Products, About, Sign In, and Register links should be present throughout all the respective pages in the application until logging in.</p> <p>** Kindly refer to the screenshots for any clarifications. **</p>
US_02	User's Page (When logged in as a User)	<p>As a logged-in user, I should be able to view the list of available products and perform actions like viewing, editing, deleting, or adding new products.</p> <p>Acceptance criteria:</p> <ol style="list-style-type: none"> 1. There should be 2 routes/links: <ul style="list-style-type: none"> ● Products: When clicking this link, it should navigate to the home page displaying the product list. ● About: When clicking this link, it should navigate to the "About" page, providing details about the application.

		<ol style="list-style-type: none"> 2. Logout: Clicking the logout button should log the user out and redirect them to the Home page. 3. You need to use the already implemented footer component to ensure that "© 2024 Products Inventory" is displayed on all pages, including after logging out. <p>The above-mentioned 2 navigation routes/links, logout button, footer should be present and functional across all the respective pages of the Product Inventory System, allowing easy access to any section from anywhere within the app until logging out.</p> <ol style="list-style-type: none"> 4. The Home Page should display the title: "Product List" as heading in h2. 5. A search bar should be available to filter the products. 6. A list of products should be displayed, where each product includes: <ul style="list-style-type: none"> ● Product Name (e.g., Laptop, Smartphone). ● Action Buttons: <ol style="list-style-type: none"> 1. View: Navigates to the Product Details page for the selected product. 2. Edit: Navigates to the Update Product page for the selected product. 3. Delete: Deletes the product from the list. 7. Add Product Button: Navigates to the Add Product page. <p>** Kindly refer to the screenshots for any clarifications. **</p> <p>View Product:</p> <p>Acceptance criteria:</p> <ol style="list-style-type: none"> 1. When clicking the "View" button, I should be able to view detailed information about a selected product. 2. It should display the title "Product Details" as heading in h2. 3. The page should show the following details for the selected product: <ul style="list-style-type: none"> ● Name: The name of the product (e.g., Laptop). ● Description: A detailed description of the product (e.g., A
--	--	--

powerful gaming laptop).

- **Manufacturer:** The manufacturer of the product (e.g., Tech Corp).
- **Price:** The price of the product (e.g., 1100).
- **Quantity:** The available quantity (e.g., 15).

**** Kindly refer to the screenshots for any clarifications. ****

Update Product:

Acceptance criteria:

1. When clicking the "Update" button, I should be able to update the details of a product.
2. It should display the title "**Update Product**" as heading in h2.
3. The page should include a form with pre-filled fields for the selected product:
 - **Name:** Editable field for the product name.
 - **Description:** Editable field for the product description.
 - **Manufacturer:** Editable field for the manufacturer name.
 - **Price:** Editable field for the product price.
 - **Quantity:** Editable field for the product quantity.
4. The form should include a **Update Product** button to update the respective products:
 - It should be **enabled** only when all required fields are filled correctly.
5. Clicking the **Update Product** button should update the product and redirect the user to the **Home Page**.

**** Kindly refer to the screenshots for any clarifications. ****

Delete Product:

Acceptance criteria:

1. When clicking the "Delete" button, it should delete the product from the list.
2. The product should be removed from the list, and the page should update to show the remaining products.

**** Kindly refer to the screenshots for any clarifications. ****

Add Product:

Acceptance criteria:

1. When clicking the "Add Product" button, I should be able to add a new product to the inventory.
2. It should display the title "**Add Product**" as heading in h2.
3. The page should include a form with the following fields:
 - **Name:** Required field for the product name. Should display a validation message if left empty as mentioned in the screenshot.
 - **Description:** Required field for the product description. Should display a validation message if left empty as mentioned in the screenshot.
 - **Manufacturer:** Required field for the manufacturer name. Should display a validation message if left empty as mentioned in the screenshot.
 - **Price:** Required field for the product price. Should display a validation message if left empty as mentioned in the screenshot.
 - **Quantity:** Required field for the product quantity. Should display a validation message if left empty as mentioned in the screenshot.
4. Add Product Button:
 - The button should be **disabled by default**.
 - It should be **enabled** only when all required fields are filled correctly.
5. Clicking the **Add Product** button should add the new product and redirect the user to the **Home Page**.

**** Kindly refer to the screenshots for any clarifications. ****

Logout Button:

1. When the "Logout" button is clicked, the user should be immediately logged out of the application.
2. Upon logout, the user should be redirected to the "Welcome" page.

		** Kindly refer to the screenshots for any clarifications. **
	About Page	<p>Acceptance criteria:</p> <ol style="list-style-type: none"> 1. When clicking the About Page link, I should be able to view the details about the application on the "About" page. 2. You need to use the already implemented About Page link, it should be able to view the details about the application on the "About" page. <p>The Products, About, Logout, and Footer should be present throughout all the respective pages in the application until logging out.</p> <p>** Kindly refer to the screenshots for any clarifications. **</p>

5 CONSTRAINTS

1. You should be able to press the "TAB" key and "SHIFT + TAB" to navigate from top field to bottom field and vice-versa.
2. All required fields must be completed with valid data.
3. When logging into the system, all fields must be filled.
4. When adding or updating a product, all fields are mandatory to be filled.
5. The system should validate invalid entries and display appropriate error messages.

6 MANDATORY ASSESSMENT GUIDELINES

1. All actions like build, compile, running application, running test cases will be through Command Terminal.
2. To open the command terminal the test takers, need to go to Application menu (Three horizontal lines at left top) -> Terminal -> New Terminal.
3. This editor Auto Saves the code.
4. If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B-command

compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.

5. These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.
6. This is a web-based application, to run the application on a browser, use the internal browser in the workspace. Click on the second last option on the left panel of IDE, you can find Browser Preview, where you can launch the application.

Note: The application will not run in the local browser

7. You can follow series of command to setup Angular environment once you are in your project-name folder:
 - a. npm install -> Will install all dependencies -> takes 10 to 15 min.
 - b. npm run start -> To compile and deploy the project in browser. You can press the <Ctrl> key while clicking on localhost:4200 to open the project in the browser -> takes 2 to 3 min.
 - c. npm run json:server -> to deploy fake rest api created with json-server -> takes 10 to 15 seconds
 - d. npm run test -> to run all test cases. **It is mandatory to run this command before submission of workspace -> takes 5 to 6 min.**
8. Once you are done with development and ready with submission, you may navigate to the previous tab and submit the workspace. It is mandatory to click on **"Submit Assessment"** after you are done with code.
9. You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.