Ex. No. 9	Model-View-Controller (MVC) using Angular JS
Date of Exercise	24/10/2025

Aim:

To develop a **Single Page Web Application (SPA)** using the **AngularJS Framework** for managing a **Product Inventory System**, where users can add, display, and delete products dynamically using **AngularJS arrays** and **ng-repeat** directive. Additionally, to demonstrate the use of **AngularJS routing** for navigating between different views in the application.

Description:

In this project, an interactive **Product Inventory System** is created using **AngularJS**, an open-source JavaScript framework based on the **Model-View-Controller (MVC)** architecture.

The application enables users to:

- **Add products** by entering product name and price.
- **Store products** in an **AngularJS array**.
- **Display products** in an HTML table using the **ng-repeat** directive for dynamic data binding.
- **Delete products** from the array and automatically update the UI in real-time.

The application also demonstrates **AngularJS Routing** to navigate between multiple pages (for example, *Home*, *Add Product*, *View Products*) without reloading the entire page — making it a true **Single Page Application (SPA)**.

Key AngularJS Features Used:

- **ng-app** – Initializes the AngularJS application.
- **ng-model** – Binds user input fields (product name and price) to the model.
- **ng-repeat** – Iterates over the product array to display data dynamically in a table.
- **ng-click** – Handles delete and add operations on button clicks.

- **$routeProvider** – Manages routing between different views or templates in the SPA.

Workflow:

1. User enters a product name and price in the input fields.
2. Clicking **"Add Product"** pushes the product details into an AngularJS array.
3. The **HTML table** updates automatically to reflect the current state of the array using ng-repeat.
4. Clicking the **"Delete"** link removes the respective product from the array and updates the UI instantly.
5. Users can navigate between pages (e.g., "Add Product" and "Product List") using **AngularJS routing**.

Program:

```
<!DOCTYPE html>

<html ng-app="inventoryApp">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Product Inventory System</title>

  <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>

  <link rel="stylesheet" href="style.css">

</head>

<body ng-controller="inventoryController">

  <div class="container">

    <!-- Header -->
```

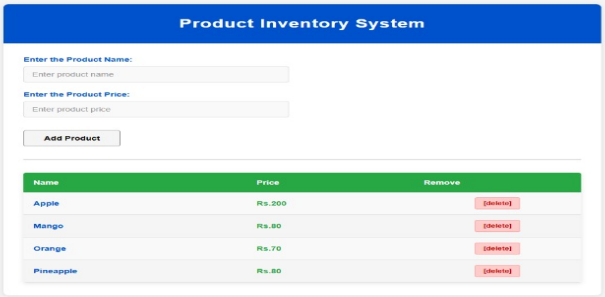
```
<div class="header">
    <h1>Product Inventory System</h1>
</div>
<!-- Main Content -->
<div class="content">
    <!-- Input Section -->
    <div class="input-section">
        <div class="form-group">
            <label for="productName">Enter the Product Name:</label>
            <input
                type="text"
                id="productName"
                ng-model="newProduct.name"
                placeholder="Enter product name"
                class="input-field">
        </div>
        <div class="form-group">
            <label for="productPrice">Enter the Product Price:</label>
            <input
                type="number"
                id="productPrice"
                ng-model="newProduct.price"
                placeholder="Enter product price"
```

```
                class="input-field"
                min="0">
            </div>
            <button ng-click="addProduct()" class="btn-add-product">Add Product</button>
        </div>
        <hr class="divider">
        <!-- Products Table Section -->
        <div class="table-section">
            <table class="products-table" ng-if="products.length > 0">
                <thead>
                    <tr>
                        <th class="col-name">Name</th>
                        <th class="col-price">Price</th>
                        <th class="col-action">Remove</th>
                    </tr>
                </thead>
                <tbody>
                    <tr ng-repeat="product in products">
                        <td class="cell-name">{{ product.name }}</td>
                        <td class="cell-price">Rs.{{ product.price }}</td>
                        <td class="cell-action">
                            <button ng-click="removeProduct($index)"
class="btn-delete">[delete]</button>
                        </td>
```

```
                </tr>
              </tbody>
            </table>
            <div ng-if="products.length === 0" class="empty-message">
              <p>No products added yet. Add a product to get started!</p>
            </div>
          </div>
        </div>
      </div>

    <script src="app.js"></script>
</body>
</html>
```

Output:



Result:

Successfully designed and implemented a web page that uses JavaScript to dynamically modify styles and create animations and executed.