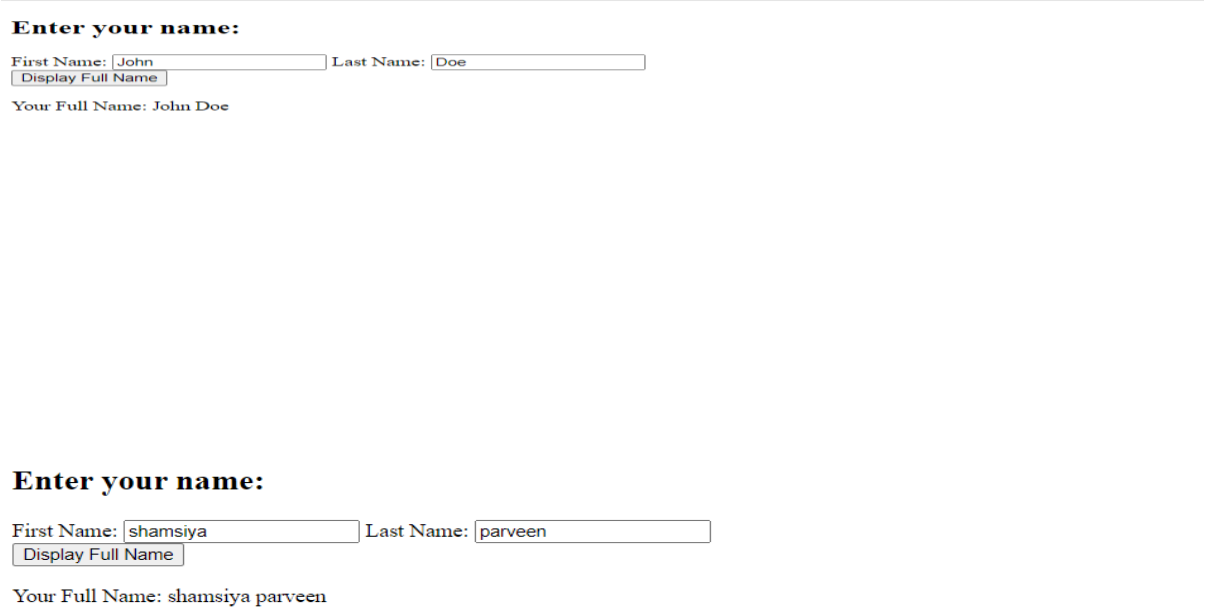
1. Develop Angular JS program that allows user to input their first name and last name and display their full name. Note: The default values for first name and last name may be included in the program.

```
<!DOCTYPE html>
<html ng-app="fullNameApp">

<head>
  <title>AngularJS Full Name App</title>
  <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>

<body ng-controller="fullNameCtrl">

  <h2>Enter your name:</h2>

  <label for="firstName">First Name:</label>
  <input type="text" ng-model="firstName" placeholder="Enter your first name">

  <label for="lastName">Last Name:</label>
  <input type="text" ng-model="lastName" placeholder="Enter your last name">

  <br>

  <button ng-click="displayFullName()">Display Full Name</button>

  <br>

  <p ng-show="fullName">Your Full Name: {{ fullName }}</p>

  <script>
    var app = angular.module('fullNameApp', []);

    app.controller('fullNameCtrl', function ($scope) {
      // Default values
      $scope.firstName = 'John';
      $scope.lastName = 'Doe';

      $scope.displayFullName = function () {
        // Concatenate first name and last name
        $scope.fullName = $scope.firstName + ' ' + $scope.lastName;
      };
    });
  </script>

</body>

</html>
```

OUTPUT:



In this Application Program, the AngularJS code creates an app named fullNameApp with a controller named fullNameCtrl. The controller has default values for first name and last name, and a function displayFullName is defined to concatenate the first name and last name and assign it to the $scope.fullName variable. The HTML contains input fields for the first name and last name, a button to trigger the display function, and a paragraph to show the full name.

2. Develop an Angular JS application that displays a list of shopping items. Allow users to add and remove items from the list using directives and controllers.Note: The default values of items may be included in the program.

```
<!DOCTYPE html>
<html ng-app="shoppingApp">

<head>
  <title>AngularJS Shopping App</title>
  <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>

<body ng-controller="shoppingCtrl">

  <h2>Shopping List</h2>

  <ul>
    <li ng-repeat="item in shoppingItems">{{ item.name }} - <button ng-
click="removeItem($index)">Remove</button></li>
  </ul>

  <label for="newItem">Add New Item:</label>
  <input type="text" ng-model="newItem" placeholder="Enter item name">
  <button ng-click="addItem()">Add Item</button>

  <script>
    var app = angular.module('shoppingApp', []);

    app.controller('shoppingCtrl', function ($scope) {
      // Default shopping items
      $scope.shoppingItems = [
        { name: 'Milk' },
        { name: 'Bread' },
        { name: 'Eggs' }
      ];

      // Function to add a new item
      $scope.addItem = function () {
        if ($scope.newItem) {
          $scope.shoppingItems.push({ name: $scope.newItem });
          $scope.newItem = ''; // Clear input after adding item
        }
      };

      // Function to remove an item
      $scope.removeItem = function (index) {
        $scope.shoppingItems.splice(index, 1);
      };
    });
  </script>
```

</body>

</html>

output

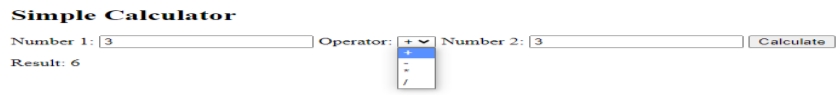
In this Application Program, the AngularJS code creates an app named shoppingApp with a controller named shoppingCtrl. The controller has a default list of shopping items and functions to add and remove items. The HTML contains an unordered list (<ul>) that uses ng-repeat to loop through the shopping items and display them. There's also an input field and a button to add new items, and each item has a "Remove" button to remove it from the list.

3. Develop a simple Angular JS calculator application that can perform basic mathematical operations (addition, subtraction, multiplication, division) based on user input.

```
<!DOCTYPE html>
<html lang="en" ng-app="calculatorApp">
<head>
   <meta charset="UTF-8">
   <title>AngularJS Calculator</title>
   <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>
<body>

<div ng-controller="CalculatorController">
   <h2>Simple Calculator</h2>

   <label for="num1">Number 1:</label>
   <input type="number" ng-model="num1" id="num1" />

   <label for="operator">Operator:</label>
   <select ng-model="operator" id="operator">
      <option value="+">+</option>
      <option value="-">-</option>
      <option value="*">*</option>
      <option value="/">/</option>
   </select>

   <label for="num2">Number 2:</label>
   <input type="number" ng-model="num2" id="num2" />

   <button ng-click="calculate()">Calculate</button>

   <p ng-if="result !== undefined">Result: {{ result }}</p>
</div>

<script>
   var app = angular.module('calculatorApp', []);

   app.controller('CalculatorController', function ($scope) {
      $scope.calculate = function () {
         var num1 = parseFloat($scope.num1);
         var num2 = parseFloat($scope.num2);

         if (isNaN(num1) || isNaN(num2)) {
            $scope.result = 'Invalid input';
            return;
         }

         switch ($scope.operator) {
```

```
            case '+':
                $scope.result = num1 + num2;
                break;
            case '-':
                $scope.result = num1 - num2;
                break;
            case '*':
                $scope.result = num1 * num2;
                break;
            case '/':
                if (num2 !== 0) {
                    $scope.result = num1 / num2;
                } else {
                    $scope.result = 'Cannot divide by zero';
                }
                break;
            default:
                $scope.result = 'Invalid operator';
        }
    };
});
</script>

</body>
</html>
```

OUTPUT:



In this Application Program,Two input fields (num1 and num2) are used to take numeric input from the user.
A dropdown (operator) is used to select the mathematical operation (addition, subtraction, multiplication, division).
The calculate function is invoked when the user clicks the "Calculate" button, which performs the selected operation based on user input and displays the result.
Basic input validation is performed to ensure that the user inputs valid numeric values and to avoid division by zero.
This is a basic example, and you can enhance it further based on your specific requirements.