DEPARTMENT OF ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

THE OXFORD COLLEGE ENGINEERING

Hosur Road, Bommanahalli, Bengaluru-560 068

Website:www.theoxford.edu/http://theoxfordengg.org/Email :hodaiml@theoxford.edu
APPROVED BY AICTE, ACCREDITED BY NAAC, AFFILIATED TO VTU

LAB MANUAL FOR

ANGULAR JS
21CSL581

Vision of the Institute:

“To be a Respected and Most Sought after Engineering Educational Institution Engaged in Equipping Individuals capable of Building Learning Organizations in the New Millennium”.

Mission of the Institute:

Our Mission is to develop Competent Students with good value Systems and Face Challenges of the Continuously Changing World.

Vision of the Department:

“To Create Technocrats with Cognitive Skills and Technical Proficiency to Succeed in the Challenging World of New Era”.

Mission of the Department:

MD1	**To Produce outstanding Artificial Engineering Professionals with cognitive skills.**
MD2	**To Enrich the students’ skill set by continuous learning and research capabilities with vibrant ambience.**
MD3	**To empower students with Technical Proficiency, Competency and Ethicalness for the new Era.**

Program Educational Objectives (PEOs)

PEO 1	***To* Empower *graduates with* cognitive skills *to lead their professional career in Reputed Industries and Solve Problems by Applying the Principles of Mathematics, Artificial Intelligence and Machine Learning, Scientific Investigations using the Latest Technologies through the opportunities of Artificial Intelligence & Machine Learning.***
PEO 2	***To* Enrich *the graduates by engaging them in research area of Artificial Intelligence & Machine Learning and empower them to work in scientific environment.***
PEO 3	***To create graduates with Professional Advancement, Communication Skills, Life Long Learning Process, Ethical Attitude, Social Responsibility, Team Work, Project Management and Leadership Skills Through Continuing Education.***

Program Specific Outcomes (PSOs)

PSO 1	***Use appropriate Techniques and Tools in application design with a knowledge of Computer Science, Networking, Software Engineering, Programs, Projects, Design of new Algorithms, Artificial Intelligence and Machine Learning Systems for Solving Complex Engineering Problems.***
PSO 2	***Inculcate the ability to work with Professional Ethics, Communication Skills, Team Work, Exchange of Innovative Ideas to Carryout Lifelong Learning with the state of art technologies and development.***

PROGRAM OUTCOMES

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

INDEX

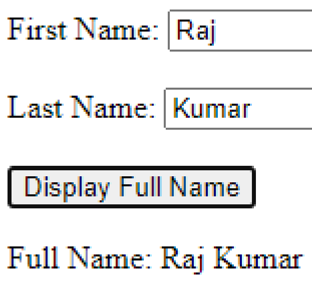
S. No.	PROGRAME NAME	PAGE No.
1	Develop Angular JS program that allows user to input their first name and last name and display their full name. Note: The default values for first name and last name may be included in the program.	6
2	Develop an Angular JS application that displays a list of shopping 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.	7
3	Develop a simple Angular JS calculator application that can perform basic mathematical operations (addition, subtraction, multiplication, division) based on user input.	8
4	Write an Angular JS application that can calculate factorial and compute square based on given user input.	9
5	Develop AngularJS application that displays a details of students and their CGPA. Allow users to read the number of students and display the count. Note: Student details may be included in the program.	10
6	Develop an AngularJS program to create a simple to-do list application. Allow users to add, edit, and delete tasks. Note: The default values for tasks may be included in the program.	11
7	Write an AngularJS program to create a simple CRUD application (Create, Read, Update, and Delete) for managing users.	13
8	Develop AngularJS program to create a login form, with validation for the username and password fields	15
9	Create an AngularJS application that displays a list of employees and their salaries. Allow users to search for employees by name and salary. Note: Employee details may be included in the program.	17
10	Create AngularJS application that allows users to maintain a collection of items. The application should display the current total number of items, and this count should automatically update as items are added or removed. Users should be able to add items to the collection and remove them as needed. Note: The default values for items may be included in the program.	18

11	Create AngularJS application to convert student details to Uppercase using angular filters. Note: The default details of students may be included in the program.	18
12	Create an AngularJS application that displays the date by using date filter parameters	19

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

```
<html ng-app="nameApp">
<head>
     <title>AngularJS Full Name Example</title>

    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.0/angular.min.js"></script>
</head>
<body>
    <div ng-controller="nameCtrl">
        <!-- Input fields for first name and last name -->
        First Name:
        <input type="text" ng-model="firstName" placeholder="Enter your first name">
        <br> <br>
        Last Name:
        <input type="text" ng-model="lastName" placeholder="Enter your last name">
        <br> <br>
        <!-- Button to display the full name -->
        <button ng-click="displayFullName()">Display Full Name</button>

        <!-- Display the full name -->
        <h1>Full Name is: {{ fullName }}</h1>
    </div>

    <script>
        angular.module('nameApp', [])
            .controller('nameCtrl', function ($scope) {
                // Default values for first name and last name
                $scope.firstName = 'Raj';
                $scope.lastName = 'Kumar';

                // Function to display the full name
                $scope.displayFullName = function () {
                    $scope.fullName = $scope.firstName + ' ' + $scope.lastName;
                };
            });
    </script>
</body>
</html>
```

Sample Output:


First Name: Raj

Last Name: Kumar

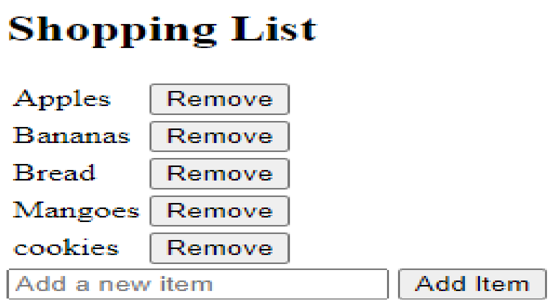
Full Name: Raj Kumar

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.

```
<html ng-app="shoppingApp">
<head>
    <title>AngularJS Shopping List</title>
    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.0/angular.min.js"></script>
</head>
<body ng-controller="shoppingCtrl">
    <h2>Shopping List</h2>
    <!-- Display the items in list
    <ul>
        <li ng-repeat="item in shoppingItems">{{ item }} &nbsp;
            <button ng-click="removeItem($index)">Remove</button>
        </li>
    </ul> -->
    <table>
         <tr ng-repeat="item in shoppingItems">
              <td>{{ item }}</td>
              <td><button ng-click="removeItem($index)">Remove</button></td>
          </tr>
       </table>
    <!-- Input field and button to add a new item -->
    <input type="text" ng-model="newItem" placeholder="Add a new item">
    <button ng-click="addItem()">Add Item</button>

    <script>
        angular.module('shoppingApp', [])
            .controller('shoppingCtrl', function ($scope) {
                // Default values for shopping items
                $scope.shoppingItems = ['Apples', 'Bananas', 'Bread', 'Milk'];

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

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

Sample Output:



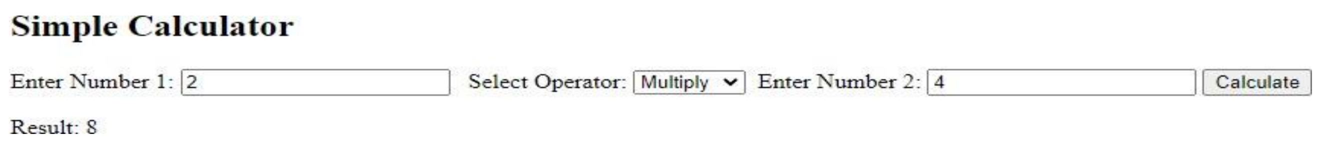
Shopping List

```
Apples   Remove
Bananas  Remove
Bread    Remove
Mangoes  Remove
cookies  Remove
Add a new item    Add Item
```

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

```
        <html ng-app="calculatorApp">
        <head>
          <title>AngularJS Calculator</title>
          <script
        src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>
        <body ng-controller="calculatorController">
          <h2>Simple Calculator</h2>
          Enter Number 1:
          <input type="number" ng-model="num1" /> &nbsp;
          Select Operator:
          <select ng-model="operator">
            <option value="+">Add</option>
            <option value="-">Subtract</option>
            <option value="*">Multiply</option>
            <option value="/">Divide</option>
          </select>&nbsp;
          Enter Number 2:
          <input type="number" ng-model="num2" />
          <button ng-click="calculate()">Calculate</button>
          <p ng-show="result !== undefined">Result: {{ result }}</p>
          <script>
            var app = angular.module('calculatorApp', []);
            app.controller('calculatorController', function ($scope) {
              $scope.calculate = function () {
                switch ($scope.operator) {
                  case '+':
                    $scope.result = $scope.num1 + $scope.num2;
                    break;
                  case '-':
                    $scope.result = $scope.num1 - $scope.num2;
                    break;
                  case '*':
                    $scope.result = $scope.num1 * $scope.num2;
                    break;
                  case '/':
                    if ($scope.num2 !== 0) {
                      $scope.result = $scope.num1 / $scope.num2;
                    } else {
                      $scope.result = 'Cannot divide by zero';
                    }
                    break;

                }
              };
            });
          </script>
        </body>
        </html>
```

Sample Output:



Simple Calculator

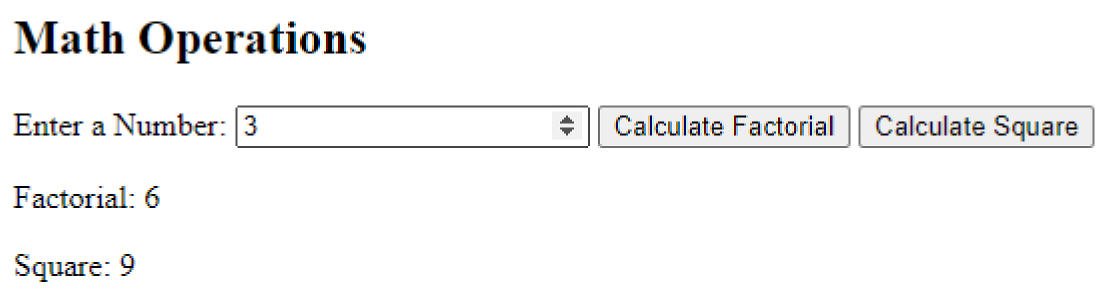
Enter Number 1: 2 Select Operator: Multiply Enter Number 2: 4 Calculate

Result: 8

4. Write an Angular JS application that can calculate factorial and compute square based on given user input.

```
<html ng-app="mathApp">
<head>
  <title>AngularJS Math Operations</title>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>
<body ng-controller="mathController">
  <h2>Math Operations</h2>
  Enter a Number:
  <input type="number" ng-model="inputNumber" />
  <button ng-click="calculateFactorial()">Calculate Factorial</button>
  <button ng-click="calculateSquare()">Calculate Square</button>

  <p ng-show="factorialResult !== undefined">Factorial: {{ factorialResult }}</p>
  <p ng-show="squareResult !== undefined">Square: {{ squareResult }}</p>
  <script>
    var app = angular.module('mathApp', []);
    app.controller('mathController', function ($scope) {
      $scope.calculateFactorial = function () {
        if ($scope.inputNumber >= 0) {
          $scope.factorialResult = factorial($scope.inputNumber);
        } else {
          $scope.factorialResult = 'Cannot calculate factorial for negative numbers';
        }
      };

      $scope.calculateSquare = function () {
        $scope.squareResult = $scope.inputNumber * $scope.inputNumber;
      };

      function factorial(n) {
        if (n == 0 || n == 1) {
          return 1;
        } else {
          return n * factorial(n - 1);
        }
      }
    });
  </script>
</body>
</html>
```

Sample Output:



Math Operations

Enter a Number: 3

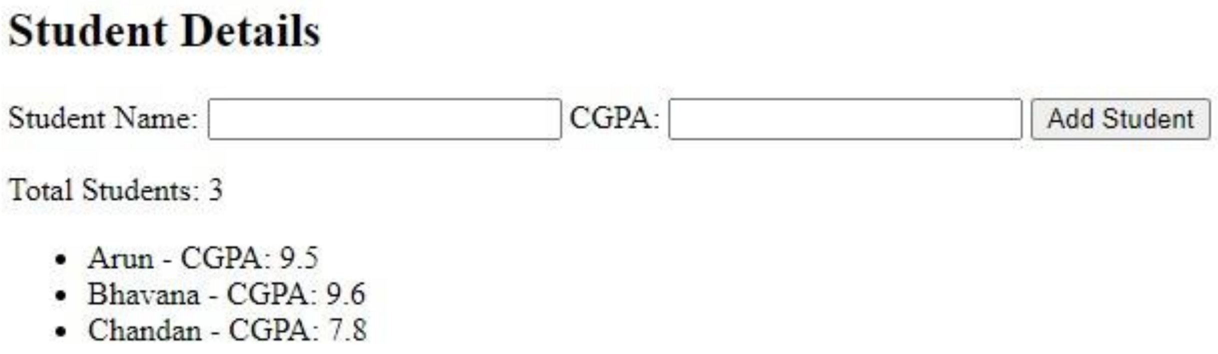
Factorial: 6

Square: 9

5. Develop AngularJS application that displays a details of students and their CGPA. Allow users to read the number of students and display the count. Note: Student details may be included in the program.

```
<html ng-app="studentApp">
<head>
  <title>AngularJS Student Details</title>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>
<body ng-controller="studentController">
  <h2>Student Details</h2>
  Student Name:
  <input type="text" ng-model="name" />
  CGPA:
  <input type="number" ng-model="cgpa" ng-min="1" ng-max="10"/>
  <button ng-click="addStudent()">Add Student</button>

  <p>Total Students: {{ students.length }}</p>

  <ul>
    <li ng-repeat="student in students">
      {{ student.name }} - CGPA: {{ student.cgpa }}
    </li>
  </ul>

  <script>
    var app = angular.module('studentApp', []);
    app.controller('studentController', function ($scope) {
      $scope.students = [];

      $scope.addStudent = function () {
        if ($scope.name && $scope.cgpa) {
          $scope.students.push({
    name: $scope.name,cgpa: $scope.cgpa
});
// Clear the input fields
$scope.name = '';
$scope.cgpa = '';

        }
      };
});

  </script>
</body>
</html>
```

Sample Output:



Student Details

Student Name: CGPA:

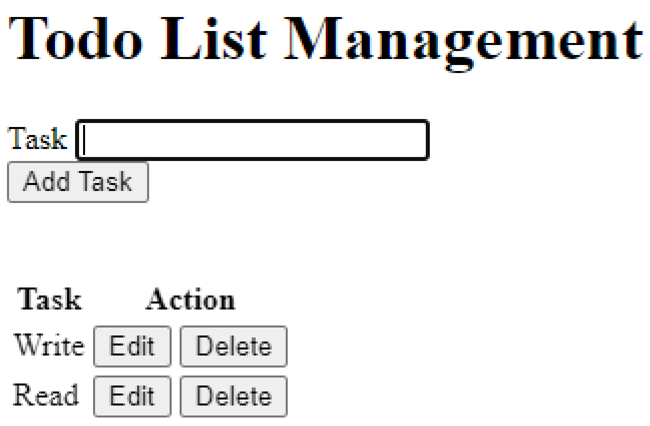
Total Students: 3

- Arun - CGPA: 9.5
- Bhavana - CGPA: 9.6
- Chandan - CGPA: 7.8

6.Develop an AngularJS program to create a simple to-do list application. Allow users to add, edit, and deletetasks.Note: The default values for tasks may be included in the program.

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

<head>
        <title>AngularJS Todo List</title>
        <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>
<body ng-controller="todoController">
        <h1>Todo List</h1>

        <!-- Form for adding a new task -->
        <form ng-submit="addTask()">
                Task:
                <input type="text" ng-model="newTask" required>
                <button type="submit">Add Task</button>
        </form>
        <br>

        <!-- Table to display task information -->
        <table>
                <thead>
                        <tr>
                                <th>Task</th>
                                <th>Action</th>
                        </tr>
                </thead>
                <tbody>
                        <tr ng-repeat="task in tasks">
                                <td>{{ task }}</td>
                                <td>
                                        <button ng-click="editTask($index)">Edit</button>
                                        <button ng-click="deleteTask($index)">Delete</button>
                                </td>
                        </tr>
                </tbody>
        </table>

        <!-- Edit Task Modal -->
        <div ng-if="editingTaskIndex !== null">
                <h2>Edit
                Task</h2>Task:
                <input type="text" ng-model="tasks" required>
                <br>
                <button ng-click="saveEdit()">Save</button>
                <button ng-click="cancelEdit()">Cancel</button>
        </div>

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

                app.controller('todoController', function ($scope) {
                        $scope.tasks =
                                ['Task 1',
                                'Task 2',
                                'Task 3'
];

                        $scope.newTask = '';
                        $scope.editingTaskIndex = null;
                        $scope.addTask = function () {
                                $scope.tasks.push($scope.newTask);
                                $scope.newTask = '';
                        };
$scope.editTask = function (index) {
                        // Prompt for updated task with validation
                        var updatedTask = prompt('Enter updated task:');

                        // Check if the user pressed cancelif
                        (updatedTask !== null) {
                                // Update the task
                                $scope.tasks.splice(index, 1, updatedTask);
                        }
                };


                              $scope.deleteTask = function (index) {
                                  $scope.tasks.splice(index, 1);
                        };
                });
        </script>
</body>

</html>
```

Sample Output:


Todo List Management

Task

Add Task

Task	Action
Write	Edit Delete
Read	Edit Delete

7. Write an AngularJS program to create a simple CRUD application (Create, Read, Update, and Delete) for managing users.

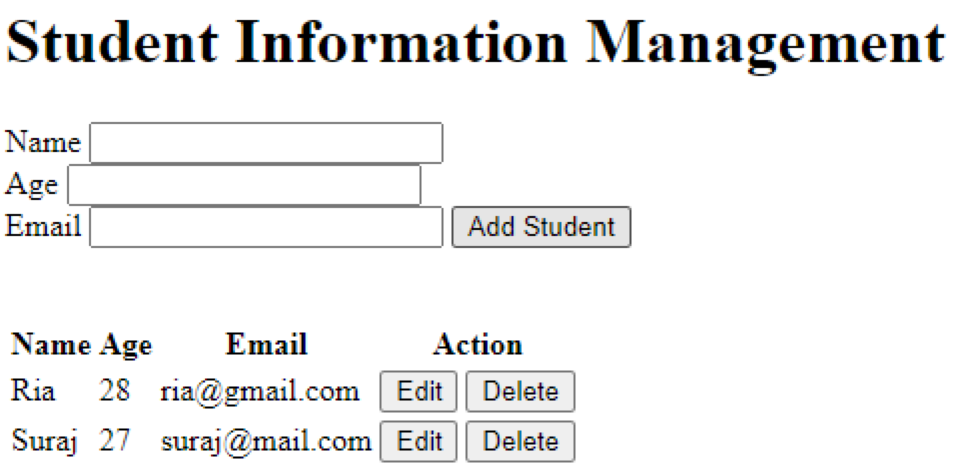
```
<!DOCTYPE html>
<html ng-app="crudApp">

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

<body ng-controller="crudController">
        <h1>User Management</h1>

        <!-- Form for adding a new user -->
        <form ng-submit="addUser()">
                Name:
```

```
          <input type="text" ng-model="name" required>
          <
          b
          r
          A
          g
          e
          :
          <input type="number" ng-model="age" required>
          <br>
          <button type="submit">Add User</button>
      </form>
      <br>

      <!-- Table to display user information -->
      <table>
          <thead>
              <tr>
                  <th>Name</th>
                  <th>Age</th>
                  <th>Action</th>
              </tr>
          </thead>
          <tbody>
              <tr ng-repeat="user in users">
                  <td>{{ user.name }}</td>
                  <td>{{ user.age }}</td>
                  <td>
                      <button ng-click="editUser(user)">Edit</button>
                      <button ng-click="deleteUser(user)">Delete</button>
                  </td>
              </tr>
          </tbody>
      </table>

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

          app.controller('crudController', function ($scope) {
              $scope.users = [
                  { name: 'Ram', age: 25 },
                  { name: 'Sam', age: 30 },
              ];

              $scope.addUser = function () {
                  $scope.users.push({ name: $scope.name, age: $scope.age });
                  $scope.name = '';
$scope.age = '';
```

```
              };

              $scope.editUser = function (user) {
                  var index = $scope.users.indexOf(user);

                  // Prompt for updated values with validation
                  var updatedName = prompt('Enter updated name:', user.name);var updatedAge
                  = prompt('Enter updated age:', user.age);

//Check if the user pressed cancel
                  if (!(updatedName == null && updatedAge == null) ){
                      // Update the user
                      var updatedUser = { name: updatedName, age: parseInt(updatedAge)
    };
                      $scope.users.splice(index, 1, updatedUser);
```

```
                    }
                };

                $scope.deleteUser = function (user) {
                    var index = $scope.users.indexOf(user);
                    $scope.users.splice(index, 1);
                };
            });
        </script>
    </body>

    </html>
```

Sample Output:



Student Information Management

Name	
Age	
Email	

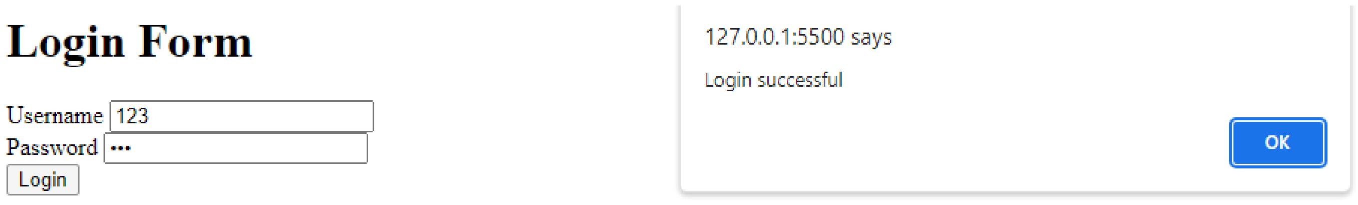
Add Student

Name	Age	Email	Action
Ria	28	ria@gmail.com	Edit Delete
Suraj	27	suraj@mail.com	Edit Delete

8.Develop AngularJS program to create a login form, with validation for the username and password fields.

```
    <html ng-app="loginApp">
        <script
    src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
<body ng-controller="loginController">
<h1>Login Form</h1>
        <!-- Form for login with validation -->
        <form ng-
            submit="login()"
            >Username
            <input type="text" ng-model="username" required>
            <b
            r>
            Pa
            ss
            wo
            rd
            <input type="password" ng-model="password" required>
            <br>
            <button type="submit">Login</button>
        </form>

        <script>
            var app = angular.module('loginApp', []);
            app.controller('loginController', function
            ($scope) {
                    $scope.login = function () {

                    // Check if username is "Ram" and password is "Ram"
                    if ($scope.username == 'ram' && $scope.password ==
                        'ram') {alert('Login successful');
                        // Add further logic for successful login
                    } else {

                        alert('Login failed. Invalid username or password.');
                        // Add logic for failed login

                    }
                };
            });
        </script>

    </body>
    </html>
```

Sample Output:


Login Form

Username 123
Password •••
Login

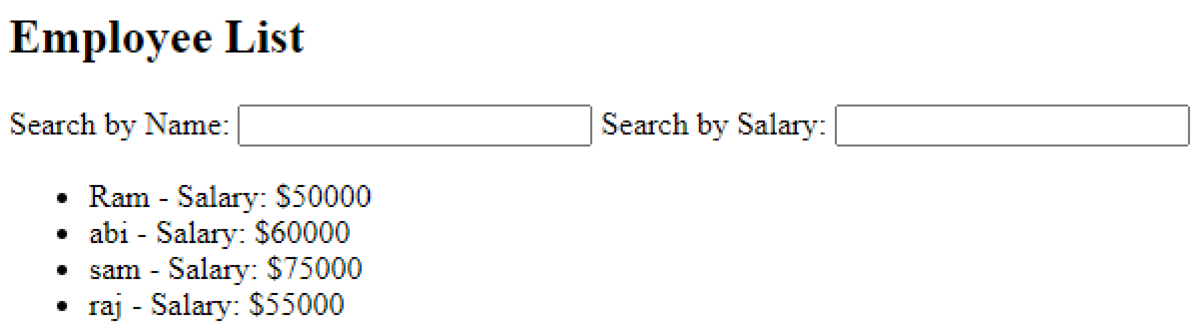
127.0.0.1:5500 says
Login successful
OK

9.Create an AngularJS application that displays a list of employees and their salaries. Allow users to search for employees by name and salary. Note: Employee details may be included in the program.

```
<html ng-app="employeeApp">
<head>
  <title>AngularJS Employee Search</title>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
 </head>
<body ng-controller="employeeController">
  <h2>Employee
List</h2>Search by
Name:
  <input type="text" ng-model="searchName" />

Search by Salary:
  <input type="number" ng-model="searchSalary" />

  <ul>
    <li ng-repeat="employee in employees | filter: {name: searchName,
salary:searchSalary}">
      {{ employee.name }} - Salary: Rs{{ employee.salary }}
    </li>
  </ul>

  <script>
    var app = angular.module('employeeApp', []);
    app.controller('employeeController', function
    ($scope) {
      $scope.employees = [
        { name: 'Ram', salary: 50000 },
        { name: 'abi', salary: 60000 },
        { name: 'sam', salary: 75000 },
        { name: 'raj', salary: 55000 }
      ];

      $scope.searchName = '';
```

```
            $scope.searchSalary = '';

          });
        </script>
      </body>

      </html>
```

Sample Output:



Employee List

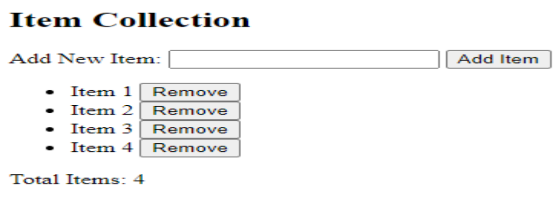
Search by Name: [ ] Search by Salary: [ ]

- Ram - Salary: $50000
- abi - Salary: $60000
- sam - Salary: $75000
- raj - Salary: $55000

10.Create Angular JS application that allows users to maintain a collection of items. The application should display the current total number of items, and this count should automatically update as items are added or removed. Users should be able to add items to the collection and remove them as needed. Note: The default values for items may be included in the program.

```
     <html ng-app="itemApp">
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
<body ng-controller="itemController">
               <h2>Item Collection</h2>
               Add New Item:
               <input type="text" ng-model="newItem" />
               <button ng-click="addItem()">Add Item</button>
               <ul>

<li ng-repeat="item in items track by $index">
  {{ item }}
  <button ng-click="removeItem($index)">Remove</button>
</li>
</ul>
               <p>Total Items: {{ items.length }}</p>
               <script>
                    var app = angular.module('itemApp', []);
                    app.controller('itemController', function ($scope) {
                      $scope.items = ['Item 1', 'Item 2', 'Item 3']; // Default items
                      $scope.newItem = '';

                      $scope.addItem =
                            function () {if
                            ($scope.newItem) {
                              $scope.items.push($scope.newItem);
                              $scope.newItem = ''; // Clear the input field
                            }
                      };

                      $scope.removeItem = function (index) {
                            $scope.items.splice(index, 1);
                 };
               });
          </script>
              </body>
              </html>
```

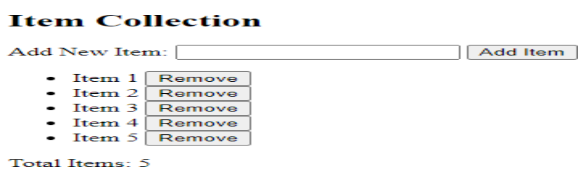
Sample Output:


Item Collection

Add New Item: Add Item

- Item 1 Remove
- Item 2 Remove
- Item 3 Remove
- Item 4 Remove

Total Items: 4


Item Collection

Add New Item: Add Item

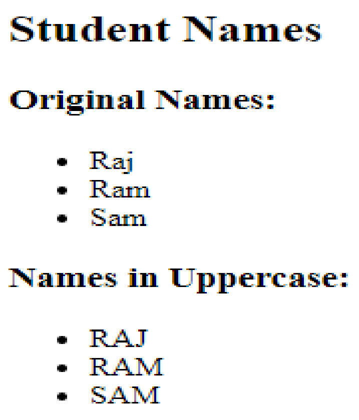
- Item 1 Remove
- Item 2 Remove
- Item 3 Remove
- Item 4 Remove
- Item 5 Remove

Total Items: 5

11.Create Angular JS application to convert student details to uppercase using angular filters. Note: The default details of students may be included in the program.

```
        <html ng-app="studentApp">
                                    <title>Student Name Converter</title>
                                    <script
        src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>

<body ng-controller="studentController">
                                    <h2>Student Names</h2>

                                    <!-- Display the original student names -->
                                    <h3>Original Names:</h3>
                                    <ul>
                                              <li ng-repeat="name in names">
                                                        {{ name }}
                          </li>
                    </ul>


                    <!-- Display the student names in uppercase using filters -->
                    <h3>Names in Uppercase:</h3>
                    <ul>
                        <li ng-repeat="name in names">
                             {{ name | uppercase }}
                      </li>
                </ul>

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

                          app.controller('studentController', function ($scope) {
                                $scope.names = ['Raj', 'Ram', 'Sam'];
                          });
                    </script>
        </body>
        </html>
```

Sample Output:


Student Names

Original Names:

- Raj
- Ram
- Sam

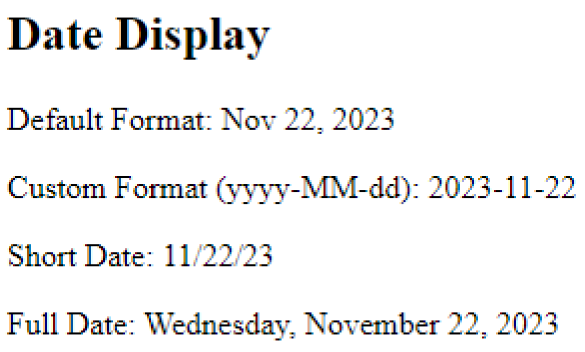
Names in Uppercase:

- RAJ
- RAM
- SAM

12. Create an AngularJS application that displays the date by using date filter parameters

```
	<!DOCTYPE html>
	<html ng-app="dateApp">
	<head>
		<title>Date Display Application</title>
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>
	<body ng-controller="dateController">

		<h2>Date Display</h2>

		<!-- Display the current date with various filter parameters -->
		<p>Default Format: {{ currentDate | date }}</p>
		<p>Custom Format (yyyy-MM-dd): {{ currentDate | date:'yyyy-MM-dd' }}</p>
		<p>Short Date: {{ currentDate | date:'shortDate' }}</p>
		<p>Full Date: {{ currentDate | date:'fullDate' }}</p>

		<script>
			var app = angular.module('dateApp', []);
			app.controller('dateController', function ($scope) {
				$scope.currentDate = new Date();
			});
		</script>
	</body>
	</html>
```

Sample Output:



```
Date Display

Default Format: Nov 22, 2023

Custom Format (yyyy-MM-dd): 2023-11-22

Short Date: 11/22/23

Full Date: Wednesday, November 22, 2023
```