CAMBRIDGE INSTITUTE OF TECHNOLOGY

Affiliated to VTU, Recognized by GOK, Approved by AICTE, New Delhi (NAAC 'A+ Grade' Accredited, NBA Accredited (UG - CSE, ECE, ISE, EIE and EEE) K R Puram, Bengaluru - 560 036 Ph:(080)28611880,28611881

Department of Artificial Intelligence & Machine Learning

ANGULAR JS LABORATORYMANUAL
(21CSL581/ 21CBL583)

V Semester

By:-

Dr.Buddesab
Associate Professor
Dept. of AIML

Prof. Anusha
Assistant Professor
Dept. of AIML

ANGULAR JS			
Course Code	**21CSL581/21CBL583**	CIE Marks	50
Teaching Hours/Week(L:T:P:S)	0:0:2:0	SEE Marks	50
Credits	01	Total marks	100
Examination type(SEE)	PRACTICAL		

Courseobjectives:

- To learn the basics of Angular JS framework.
- To understand the Angular JS Modules,Forms,inputs,expression,databindings and Filters
- Togainexperienceofmoderntoolusage(VSCode,Atomoranyother]indeveloping Web applications

Sl.NO	Experiments
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.
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
3	Develop a simple Angular JS calculator application that can perform basic mathematical operations (addition, subtraction, multiplication, division) based on user input.
4	Write an Angular JS application that can calculate factorial and compute square based on given user input.
5	Develop Angular JS 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
6	Develop an Angular JS 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.
7	Write an Angular JS program to create a simple CRUD application (Create, Read, Update, and Delete) for managing users.
8	Develop Angular JS program to create a login form, with validation for the username and password fields.
9	Create an Angular JS 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.
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.
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.
12	Create an Angular JS application that displays the date by using date filter parameters

NOTE:IncludenecessaryHTMLelementsandCSSfortheaboveAngularapplications.

Course out comes(CourseSkillSet):

Attheend of the course the student will be able to:

1. Develop Angular JS programs using basic features
2. Develop dynamic Web applications using Angular JS modules
3. Make use of form validations and controls for interactive applications
4. AppytheconceptsofExpressions,databindingsandfiltersindevelopingAngularJSprograms
5. MakeuseofmoderntoolstodevelopWebapplications

AssessmentDetails(bothCIEandSEE)

The weightage of Continuous Internal Evaluation (CIE) is 50% and for Semester End Exam (SEE) is 50%. Theminimum passing mark for the CIE is 40% of the **maximum** marks (20 marks). A student shall be deemed tohavesatisfiedtheacademicrequirementsandearnedthecreditsallottedtoeachcourse.Thestudenthastosecurenot less than 35% (18 Marks out of 50) in the semester-end examination (SEE). The student has to secure aminimum of 40% (40 marks out of 100) in the sum totaloftheCIE(ContinuousInternalEvaluation)andSEE(SemesterEndExamination)takentogether.

ContinuousInternalEvaluation(CIE):

CIEmarksforthepracticalcourseis**50Marks.**

Thesplit-upofCIEmarksforrecord/journalandtestareintheratio**60:40.**

- Each experiment to be evaluated for conduction with observation sheet and record write-up. Rubrics for theevaluationofthejournal/write-upforhardware/softwareexperimentsdesignedbythefacultywhoishandlingthelaboratorysessionandismadeknowntostudentsatthe beginningofthepracticalsession.
- Record should contain all the specified experiments in the syllabus and each experiment write-up will beevaluated for10marks.
- Totalmarksscoredbythestudentsarescaleddownedto30marks(60%ofmaximummarks).
- Weightagetobe givenfor neatnessandsubmissionofrecord/write-upontime.
- Department shall conduct 02 tests for 100 marks, the first test shall be conducted after the 8t^ week of thesemesterandthesecondtestshallbeconducted afterthe14'hweekof thesemester.
- Ineachtest,testwrite-up,conductionofexperiment,acceptableresult,andproceduralknowledgewillcarryaweightageof60%and therest40%forviva-voce.
- Thesuitablerubricscanbedesignedtoevaluateeachstudent'sperformanceandlearningability.Rubricssuggeste d inAnnexure-IIofRegulationbook
- Theaverageof02testsisscaleddownto**20marks**(40%of**themaximum**marks).

TheSumofscaled-downmarksscored inthereportwrite-up/journalandaveragemarksoftwo testsisthetotalCIEmarksscoredbythestudent.

SemesterEndEvaluation(SEE):

- SEEmarksforthepracticalcourseis50Marks.
- SEEshallbeconductedjointlybythetwoexaminersofthesameinstitute,examinersareappointedbytheUnive rsity
- Alllaboratoryexperimentsaretobeincludedforpracticalexamination.
- (Rubrics) Breakup of marks and the instructions printed on the cover page of the answer script tobe strictly adhered to by the examiners. OR based on the course requirement evaluation rubricsshall bedecidedjointlybyexaminers.
- Studentscanpickonequestion(experiment)fromthequestionslotpreparedbytheinternal/externalexaminersjoint ly.
- Evaluationoftestwrite-up/conductionprocedureandresult/vivawillbeconductedjointlybyexaminers.
- General rubrics suggested for SEE are mentioned here, write up -20%, Conduction procedureand result in -60%, Viva-voce 20% of maximum marks. SEE for practical shall be evaluated for 100 marks and scoredmarks shall be scaled down to 50 marks (however, based on course type, rubrics shall be decided bytheexaminers)
- ThedurationofSEE is02hours

RubricssuggestedinAnnexure-lIofRegulationbook

SuggestedLearningResources:

Textbooks

1. ShyamSeshadri,BradGreen—"AngularJS:UpandRunning:EnhancedProductivitywithStructuredWeb Apps", Apress, 0'ReillyMedia,Inc.
2. AgusKurniawan–"AngularJSProgrammingbyExample",FirstEdition,PEPress,2014

WeblinksandVideoLectures(e-Resources):

1. IntroductiontoAngularJS:https://www.youtube.com/watch?v=HEbphzK-0xE
2. AngularJSModules:https://www.youtube.com/watch?v=gWm0KmgnQkU
3. https://www.youtube.com/watch?v=zKkUN-mJtPQ
4. https://www.youtube.com/watch?v=ICl7_i2mtZA
5. https://www.youtube.com/watch?v=Y2Few_nkze0
6. https://www.youtube.com/watch?v=QoptnVCQHsU

ActivityBasedLearning(SuggestedActivitiesinClass)/PracticalBasedlearning

- Demonstrationofsimpleprojects/applications(courseproject)

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 in the 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

Display Full Name

Full Name: Raj Kumar

2. **Develop an Angular JS application that displays a list of shopping items. Allow users to add and removeitems from the list using directives and controllers.Note: The default values of items may be included inthe 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 }}  
            <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" />  
  Select Operator:
  <select ng-model="operator">
    <option value="+">Add</option>
    <option value="-">Subtract</option>
    <option value="*">Multiply</option>
    <option value="/">Divide</option>
  </select> 
  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

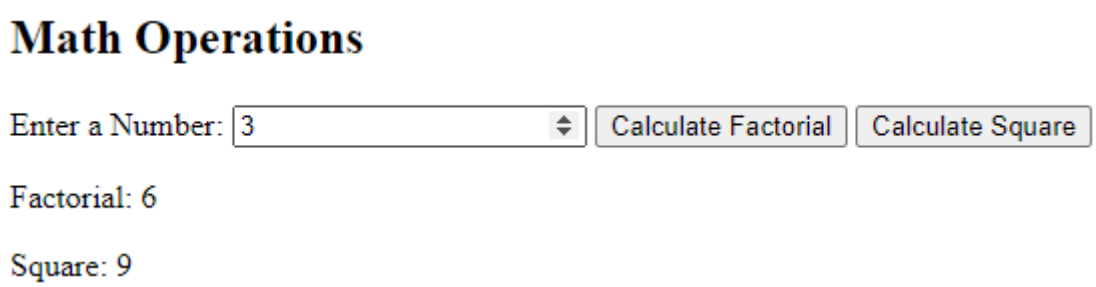
Enter Number 1: 2 Select Operator: Multiply Enter Number 2: 4 Calculate

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>
```

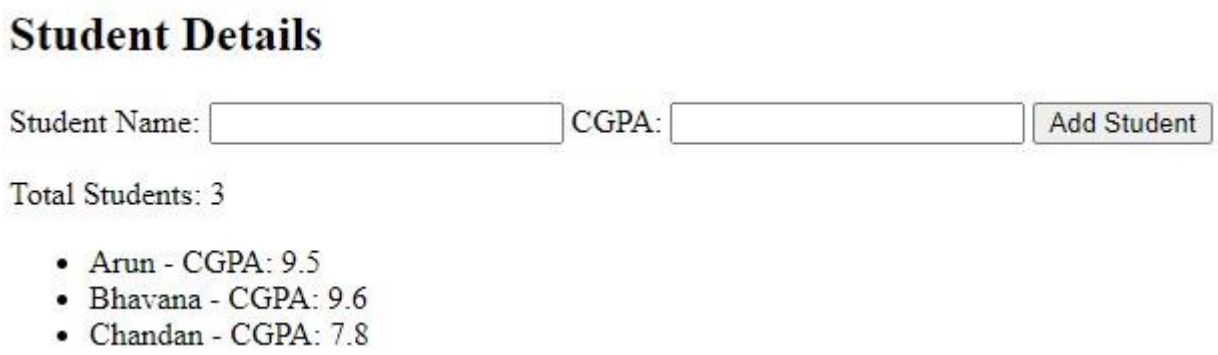
Sample Output:



5. Develop AngularJS application that displays a details of students and their CGPA. Allow users to read thenumber 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:



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 cancel
            if (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) formanaging 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>
        <br>
        Age:
        <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

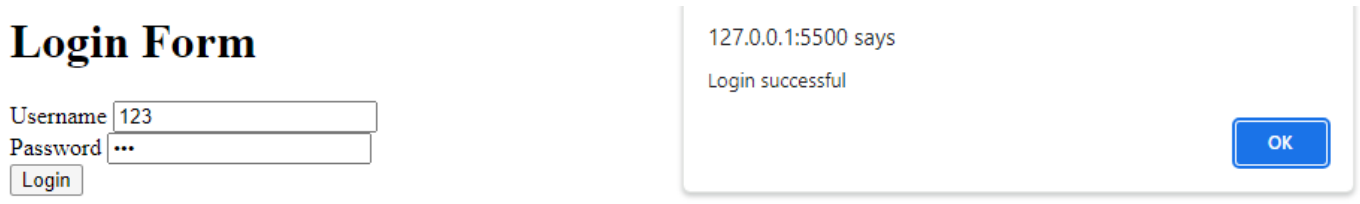
Name
Age
Email
Add Student

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

8. DevelopAngularJS 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>
        <br>
        Password
        <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:



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

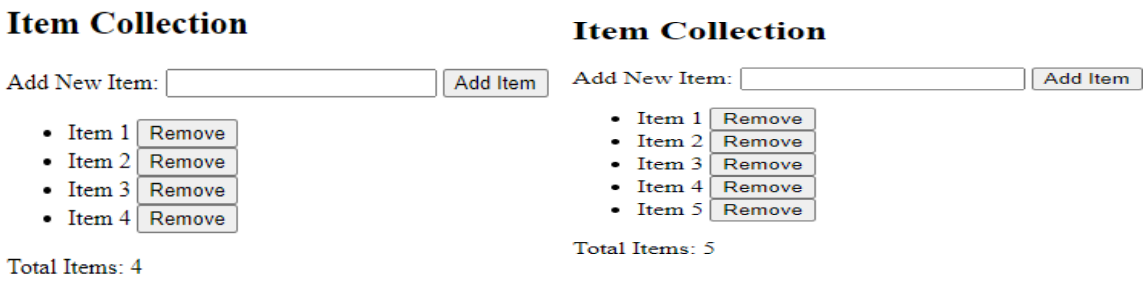
```
<html ng-app="employeeApp">
<head>
  <title>AngularJS Employee Search</title>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
</head>

<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:

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:



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