RN SHETTY TRUST®
RNS 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)
Channasandra, Dr. Vishnuvardhan Road, Bengaluru - 560 098
Ph:(080)28611880,28611881 URL: www.rnsit.ac.in
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Estd : 2001

ANGULAR JS LABORATORY MANUAL

For Fifth Semester B.E-2021 Batch
[VTU/CBCS, 2021 syllabus]

Subject Code – 21CSL581

NAME :_____

USN :_____

SECTION:_____ BATCH:_____

VISION AND MISSION OF INSTITUTION

Vision

Building RNSIT into a World Class Institution

Mission

To impart high quality education in Engineering, Technology and Management with a Difference, Enabling Students to Excel in their Career by

1. Attracting quality Students and preparing them with a strong foundation in fundamentals to achieve distinctions in various walks of life leading to outstanding contributions.
2. Imparting value based, need based, choice based, and skill based professional education to the aspiring youth and carving them into disciplined, World class Professionals with social responsibility.
3. Promoting excellence in Teaching, Research and Consultancy that galvanizes academic consciousness among Faculty and Students
4. Exposing Students to emerging frontiers of knowledge in various domains and make them suitable for Industry, Entrepreneurship, Higher studies, and Research & Development
5. Providing freedom of action and choice for all the Stake holders with better visibility

VISION AND MISSION OF CSE DEPARTMENT

Vision

Preparing better computer professionals for a real world

Mission

The Department of Computer Science and Engineering will make every effort to promote an intellectual and an ethical environment in which the strengths and skills of Computer Professionals will flourish by

1. Imparting Solid foundations and Applied aspects in both Computer Science Theory and Programming practices
2. Providing Training and encouraging R&D and Consultancy Services in frontier areas of Computer Science with a Global outlook
3. Fostering the highest ideals of Ethics, Values and creating Awareness on the role of Computingin Global Environment
4. Educating and preparing the graduates, highly Sought-after, Productive, and Well-respected fortheir work culture
5. Supporting and inducing Lifelong Learning practice

ANGULAR JS LABORATORY
INTERNAL EVALUATION SHEET

EVALUATION (MAX MARKS 20)			
TEST **A**	**REGULAR EVALUATION** **B**	**RECORD** **C**	**TOTAL MARKS** **A+B+C**
20	**20**	**10**	**50**

R1: REGULAR LAB EVALUATION WRITE UP RUBRIC (MAX MARKS 10)				
Sl. No.	**Parameters**	**Good**	**Average**	**Needs improvement**
a.	**Understanding of problem (2 marks)**	Clear understanding of problem statement while designing and implementing the program (2)	Problem statement is understood clearly but few mistakes while designing and implementing program (1)	Problem statement is not clearly understood while designing the program (1)
b.	**Writing program (2 marks)**	Program handles all possible conditions (2)	Average condition is defined and verified. (1)	Program does not handle possible conditions (1)
c.	**Design, implementation, and demonstration (3 marks)**	Program follows syntax and semantics of C programming language. Demonstrates the complete knowledge of the program written (3)	Program has few logical errors, moderately demonstrates all possible concepts implemented in programs (2)	Syntax and semantics of AngularJS is not clear (1)
d.	**Result and documentation (3 marks)**	Meticulous documentation and all conditions are taken care (3)	Acceptable documentation shown (2)	Documentation does not take care all conditions (1)

R2: REGULAR LAB EVALUATION VIVA RUBRIC (MAX MARKS 10)					
Sl. No.	**Parameter**	**Excellent**	**Good**	**Average**	**Needs Improvement**
a.	**Conceptual understanding & Additional Programming (10 marks)**	Answer 80% of the viva questions and execution of Additional programs listed (10)	Answers 60% of the viva questions and execution of Additional programs listed (7)	Answers 30% of the viva questions and execution of Additional programs listed(4)	Unable to relate the concepts (1)

R3: RECORD EVALUATION RUBRIC (MAX MARKS 10)				
Sl. No.	**Parameters**	**Excellent**	**Good**	**Needs Improvement**
a.	**Documentation (10 marks)**	Meticulous record writing including program, comments and as per the guidelines mentioned (10)	Write up contains program, but comments are not included (8)	Write up contains only program (6)

Test 1 (8th Week) (50 Marks)			**Test 2 (14th Week) (50 Marks)**		
Write up	**Execution**	**Viva**	**Write up**	**Execution**	**Viva**
10	**20**	**20**	**10**	**20**	**10**

A. TEST /LAB INTERNALS MARKS (MAX MARKS 20)

TEST #	Write up 10	Execution 20	Viva 10	Sign	Total 50	Avg. 50	Final 20
TEST-1							
TEST-2						50	20

B. REGULAR LAB EVALUATION (MAX MARKS 10)

Sl #	Date of Execution	Additional programs	Write up (10)	Exen. (10)	Viva (10)	Total 30	Teacher Signature
1							
2							
3							
4							
5							
6							
7							
8							
9							
10							
11							
12							
Total Marks			30			10	

SL.NO.	*TABLE OF CONTENTS*	*PAGE NO.*

Lesson planning / Execution plan			
	CONTENTS	***Week of Execution***	***Remarks***
	Introduction & Sample Experiments	***Week-2*** ***(03rd -09th Dec)***	
1.	Program that allows user to input their first name and last name and display Additional Programs	***Week-3*** ***(10th -16th Dec)***	
2.	Displays a list of shopping items. Allow users to add and remove items from the list Additional Programs	***Week-4&5*** ***4th-15th Dec***	
3 & 4	Calculator application that can perform basic mathematical operations (addition, subtraction, multiplication, division) based on user input. Calculate factorial and compute square based on given user input.	***Week-7*** ***17th -23rd Dec***	
4 & 5	displays a details of students and their CGPA. Allow users to read the number of students and display the count to create a simple to-do list application. Allow users to add, edit, and delete tasks.	***Week-7*** ***01st -06th Jan***	
	1st Internals	***Week-8*** ***7th -13th Jan***	
6 & 7	Create a simple CRUD application (Create, Read, Update, and Delete) for managing users. Create a login form, with validation for the username and password fields	***Week-9*** ***14th -20th Jan***	

8 & 9	Displays a list of employees and their salaries. Allow users to search for employees by name and salary	***Week-10*** ***21st -27th Jan***	
10	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	***Week-11*** ***28th - 03rd Feb***	
11	Create AngularJS application to convert student details to Uppercase using angular filters	***Week-12*** ***04th -10th Feb***	
12	Create an AngularJS application that displays the date by using date filter parameters	***Week-13*** ***11th – 17th feb***	
	II Internals	***Week-14*** ***18th – 24th feb***	

SYALLABUS

ANGULAR JS			
Course Code	**21CSL581**	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		

Course objectives:

- To learn the basics of Angular JS framework.
- To understand the Angular JS Modules, Forms, inputs, expression, data bindings and Filters
- To gain experience of modern tool usage (VS Code, Atom or any other] in developing 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 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.
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.
7	Write an AngularJS program to create a simple CRUD application (Create, Read, Update, and Delete) for managing users.
8	DevelopAngularJS program to create a login form, with validation for the username and password fields.
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.
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.

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.
12	Create an AngularJS application that displays the date by using date filter parameters
NOTE: Include necessary HTML elementsand CSS for the above Angular applications.	
Course outcomes (Course Skill Set): At the end of the course the student will be able to: 1. Develop Angular JS programs using basic features 2. Develop dynamic Web applications using AngularJS modules 3. Make use of form validations and controls for interactive applications 4. Apply the concepts of Expressions, data bindings and filters in developing Angular JS programs 5. Make use of modern tools to develop Web applications	

Assessment Details (both CIE and SEE)

The weightage of Continuous Internal Evaluation (CIE) is 50% and for Semester End Exam (SEE) is 50%. The minimum passing mark for the CIE is 40% of the **maximum** marks (20 marks). A student shall be deemed to have satisfied the academic requirements and earned the credits allotted to each course. The student has to securenot less than 35% (18 Marks out of 50) in the semester-end examination (SEE). The student has to secure a minimum of 40% (40 marks out of 100) in the sum total of the CIE (Continuous Internal Evaluation and SEE (Semester End Examination) taken together.

Continuous Internal Evaluation (CIE):

CIE marks for the practical course is **50 Marks.**

The split-up of CIE marks for record/ journal and test are in the ratio **60:40.**

- Each experiment to be evaluated for conduction with observation sheet and record write-up. Rubrics for the evaluation of the journal/write-up for hardware/software experiments designed by the faculty who is handling the laboratory session and is made known to students at the beginning of the practical session.
- Record should contain all the specified experiments in the syllabus and each experiment write-up will be evaluated for 10marks.
- Total marks scored by the students are scaled downed to 30 marks (60% of maximum marks).
- Weightage to be given for neatness and submission of record/write-up on time.
- Department shall conduct 02 tests for 100 marks, the first test shall be conducted after the 8t^ week of thesemester and the second test shall be conducted after the 14'h week of the semester.
- In each test, test write-up, conduction of experiment, acceptable result, and procedural knowledge will carrya weightage of 60% and the rest 40% for viva-voce.
- The suitable rubrics can be designed to evaluate each student's performance and learning ability.
- The average of 02 tests is scaled down to **20 marks** (40% of **the maximum marks**).

The Sum of scaled-down marks scored in the report write-up/journal and average marks of two tests is the total CIEmarks scored by the student.

Semester End Evaluation (SEE):

- SEE marks for the practical course is 50Marks.
- SEE shall be conducted jointly by the two examiners of the same institute, examiners are appointed bytheUniversity
- All laboratory experiments are to be included for practicalexamination.
- (Rubrics) Breakup of marks and the instructions printed on the cover page of the answer script to be strictly adhered to by the examiners. OR based on the course requirement evaluation rubrics shall be decided jointly byexaminers.
- Students can pick one question (experiment) from the questions lot prepared by theinternal/externalexaminers jointly.
- Evaluation of test write-up/ conduction procedure and result/viva will be conducted jointly by examiners.
- 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 scored marks shall be scaled down to 50 marks (however, based on course type, rubrics shall be decided bythe examiners)

Suggested Learning Resources:

Textbooks

1. ShyamSeshadri, Brad Green —"AngularJS: Up and Running: Enhanced Productivity with StructuredWeb Apps", Apress, 0'Reilly Media,Inc.
2. AgusKurniawan–"AngularJS Programming by Example", First Edition, PE Press, 2014

Weblinks and Video Lectures (e-Resources):

1. Introduction to Angular JS :https://www.youtube.com/watch?v=HEbphzK-0xE
2. Angular JS Modules :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

Activity Based Learning (Suggested Activities in Class)/ Practical Based learning

- Demonstration of simple projects /applications (course project)
- The duration of SEE is 02hours

Course Objectives: This course (21CSL581) will enable students to:

1. .To learn the basics of Angular JS framework.
2. To understand the Angular JS Modules, Forms, inputs, expression, data bindings and Filters.
3. To gain experience of modern tool usage (VS Code, Atom or any other] in developing Web applications

Course Outcomes : At the end of this course, students are able to:

CO1. Develop Angular JS programs using basic features.

CO2. Develop dynamic Web applications using AngularJS modules.

CO3. Make use of form validations and controls for interactive applications.

CO4.Apply the concepts of Expressions, data bindings and filters in developing Angular JS programs.

CO5. Make use of modern tools to develop Web applications.

CO-PO MATRIX

COURSE OUTCOMES	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3	PSO4
CO1	2	-	2	-	2	-	-	-	1	-	1	2	2	2	3	-
CO2	2	-	2	-	2	-	-	-	1	-	1	2	2	2	2	-
CO3	2	2	3	1	3	-	-	-	2	1	2	3	3	3	3	-
CO4	2	1	2	1	3	-	-	-	2	1	2	3	3	3	3	-
CO5	2	1	3	1	3	-	-	-	2	2	2	3	3	2	3	2

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

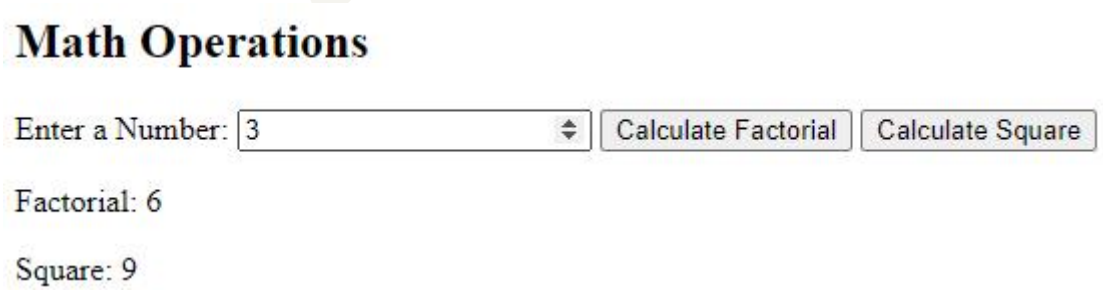
Enter Number 1: 2 Select Operator: Multiply Enter Number 2: 4 Calculate

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:

Student Details

Student Name: [ ] CGPA: [ ] Add Student

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 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

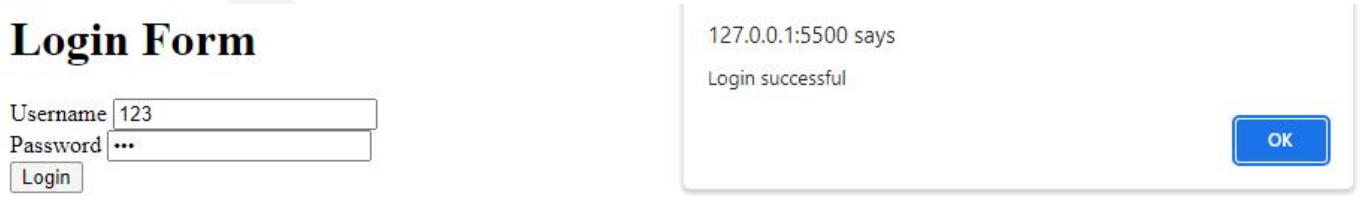
Name
Age
Email
Add Student

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

List of Additional Programs	
Angular JS	
1.	To Create an Angular JS code to Display a Message on a Web Page.
2.	To Implement Angular JS code, to display the circle's area and circumference while applying the radius as an input.
3.	To create AngularJS Code to get input from the user through a text box, clicking on the button should add to the item list and display every-time it's updated. Use ng-repeat.
4.	To create Angular JS code to Design a form that include the basic information, use Textbox,Dropdown and checkbox.
5.	To Design a nested for that includes getting name in first form, basic info in second form and password in next form, include form validation that checks for password match and length (Minimum 6 characters) using AngularJS.
6.	To Create a simple form that include the filters like converting to uppercase, lowercase, filteringbased on criteria and sorting based on some values using AngularJS.
7.	To Design a web page that includes some custom directives using Angular JS.
8.	To Design a web page that perform basic arithmetic operation using Custom Services Angular JS.
9.	To Design a Simple To-Do List using the Built-in Directive of Angular JS.
10.	Write an AngularJS Script to display list of games stored in an array on click of button using ng-click. Also demonstrate ng-init, ng-bing directives of AngularJS
11.	Write a program that demonstrates the use of AngularJS filters in an HTML.
12	Write an AngularJS script to search color name according to the character typed.
Node.js	
1.	To Build a simple HTTP server using the Node.js http module that responds with some message when accessed.
2.	To create a custom module that provides functions for basic arithmetic operations (addition,subtraction, multiplication, division) using the Node.JS.
3.	To Develop an application to render/request HTML Pages as an HTTP Server Response Using Node.JS.
4.	To build a simple application on Event emitting and handling code using Node.JS.
5.	To develop an application that performs various operation like create, read, write/append and delete the file using Node.JS.
6.	To implement an application that ask you to upload the file and store it your specified folder of the File System.
7.	To develop an Employee Database application that ADD the new employee information to thedatabase and retrieve employee information from the database using Node.JS.

8.	Write an AngularJS script to search color name according to the character typed.
9.	Create Clusters using NodeJS.
10.	Program for creating the files and read contents from the file.
11.	Uploading a file to a Node.js server
12.	Program for connecting to MySql Database using NodeJS.
13.	Program to create a form and connect to database using the node js and express js using MVC.