SAPTHAGIRI COLLEGE OF ENGINEERING

#14/5, Chikkasandra, Hesaraghatta Main Road, Bengaluru.

<u>LAB MANUAL</u>

(2023-24)ODD

21CSL581
ANGULAR JS LABORATORY
V Semester

<u>DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING</u>

Name: ____

USN: ____

Batch: ____

VISVESVARAYA TECHNOLOGICAL UNIVERSITY Jnana Sangama, Belagavi, Karnataka – 590018

21CSL581
ANGULAR JS LABORATORY
V Semester

<u>LAB MANUAL</u>

<u>Prepared By:</u>

Prof. LAVANYA K
Assistant Professor, Dept. of CSE
lavanyak@sapthagiri.edu.in

VISION AND MISSION OF INSTITUTE

VISION

To be a best institution imparting quality engineering education to deal with community needs through learning and performance.

MISSION

- To implement path breaking student centric education methods.
- To augment talent, nurture teamwork to transform to develop individual as responsible citizen.
- To educate the students and faculties about entrepreneurship to meet vibrant requirements of the society.
- Strengthen Industry-Institute Interaction for knowledge sharing.

VISION AND MISSION OF DEPARTMENT

VISION

- To be a potential quality source of expert computer engineers, addressing the requirements of industry and civic demands at large.

MISSION

- Enable learning in cutting-edge technologies adopting innovative methods.
- To enhance ability, cultivate collaboration to bring change in building students approach towards society.
- Make available platform for harnessing entrepreneurial and leadership qualities.
- Partner continuously with industry to design and implement practical syllabus.

PROGRAM EDUCATIONAL OBJECTIVES (PEO)

The program educational objectives of Bachelor of Engineering in Computer Science & Engineering at Sapthagiri College of Engineering are broadly defined on following four counts.

I. Technical expertise and problem solving

Graduates will have the expertise in analyzing real time problems and providing appropriate solutions related to Computer Science & Engineering.

II. Innovation and Research

Graduates will have the knowledge of fundamental principles and innovative technologies to succeed in higher studies and research.

III. Employability

Graduates will be advancing in their proficient skills to prepare them for improved growth in varied career paths of Computer Science and Engineering and related fields.

IV. Professional ethics

Graduates will continue to learn and to adapt technology developments combined with deep awareness of ethical responsibilities in profession.

PEO 1: Graduates will have the expertise in analyzing real time problems and providing Appropriate solutions related to Computer Science & Engineering.

PEO 2: Graduates will have the knowledge of fundamental principles and innovative technologies to succeed in higher studies, and research.

PEO 3: Graduates will continue to learn and to adapt technology developments combined with deep awareness of ethical responsibilities in profession

PROGRAM SPECIFIC OUTCOMES (PSO)

The graduates of Computer Science and Engineering program at Sapthagiri College of engineering will be able to attain the following at the time of graduation.

PSO 1: Apply the knowledge gained from Mathematics, Basic Computing, Basic Sciences and Social Sciences in general and all computer science courses in particular to identify, formulate and solve real life engineering problems.

PSO2: Identify the various analysis & design methodologies for facilitating development of high quality system software products with focus on performance optimization.

PSO3: Apply modern technology to implement in the components and its system.

PROGRAM OUTCOMES:

Engineering Graduates will be able to:

PO1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2. Problem analysis: Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3. 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.

PO4. 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.

PO5. 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.

PO6. 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.

PO7. 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.

PO8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10. 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.

PO11. 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.

PO12. 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.

DO'S AND DON'TS

Do's

- Do wear ID card and follow dress code. Do log off the computers when you finish.
- Be on time to LAB sessions.
- Do ask for assistance if you need help.
- Do keep your voice low when speaking to others in the LAB.
- Do make suggestions as to how we can improve the LAB.
- In case of any hardware related problem, ask LAB in charge for solution.
- If you are the last one leaving the LAB, make sure that the staff in charge of the LAB is informed to close the LAB
- Do keep the LAB as clean as possible.
- Do prepare for lab programs before entering the lab

Don'ts

- Do not use mobile phone inside the lab.
- Don’t do anything that can make the LAB dirty (like eating, throwing waste papers etc).
- Do not carry any external devices without permission.
- Don’t move the chairs of the LAB.
- Don’t interchange any part of one computer with another.
- Don’t leave the computers of the LAB turned on while leaving the LAB.
- Do not install or download any software or modify or delete any system files on any lab computers.
- Do not damage, remove, or disconnect any labels, parts, cables, or equipment.
- Don’t attempt to bypass the computer security system.
- Do not read or modify other user’s file.
- If you leave the lab, do not leave your personal belongings unattended.
- We are not responsible for any theft.

	Course Title:	Angular JS

	Course Code:	**21CSL581**	CIE Marks	50
	Course Type (Theory/Practical /Integrated)	Practical	SEE Marks	50
			Total Marks	100
	Teaching Hours/Week (L:T:P: S)	0:0:2:0	Exam Hours	03
	Total Hours of Pedagogy	24 hours	Credits	01

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.

Programming Exercises:

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.

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 secure not 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.**

CIE for the practical component of the IC

- 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 10 marks.
- 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 8 th week of the semester and the second test shall be conducted after the 14 th week of the semester.
- In each test, test write-up, conduction of experiment, acceptable result, and procedural knowledge will carry a 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. Rubrics suggested in Annexure-II of Regulation book
- 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 CIE marks scored by the student.

Semester End Examination (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 by the University • All laboratory experiments are to be included for practical examination.
- (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 by examiners.
- Students can pick one question (experiment) from the questions lot prepared by theinternal/external examiners 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 procedure and 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 by the examiners)
- The duration of SEE is 02hours Rubrics suggested in Annexure-lI of Regulation book

Suggested Learning Resources:

Text Books

1. ShyamSeshadri, Brad Green —"AngularJS: Up and Running: Enhanced Productivity with Structured Web 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 S :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).

COs and POs Mapping (Individual teacher has to fill up)

COs	POs						
	1	2	3	4	5	6	7
CO1							
CO2							
CO3							

Level 3- Highly Mapped, Level 2-Moderately Mapped, Level 1-Low Mapped, Level 0- Not Mapped

Program 1

AIM: 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.**

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

OUTPUT:

First Name: DUSHYANTH

Last Name: KRISHNA

Display Full Name

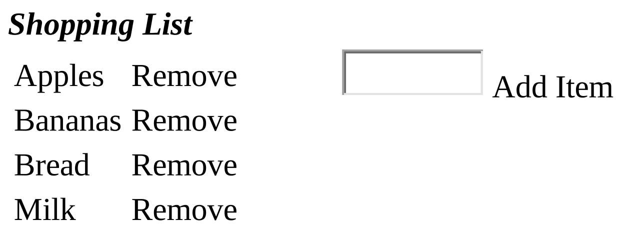
Full Name is: DUSHYANTH KRISHNA

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.

```
<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>
```

OUTPUT:



Shopping List

```
Apples   Remove          [          ] Add Item
Bananas  Remove
Bread    Remove
Milk     Remove
```

Program 3

Aim: 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>
```

OUTPUT :

Simple Calculator

Enter Number 1: 5 Select Operator: Add ; Enter Number 2: 5 Calculate

Result: 10

Program 4

Aim: 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>
```

OUTPUT :

Math Operations

```
Enter a Number:  5            Calculate Factorial          Calculate Square

Factorial: 120

Square: 25
```

Program 5

Aim: Develop Angular JS application that displays a details of students and their CGPA. Allow u sers to read the number of students and display the count. Note: Student details may be included i n the program.

```
<html>
<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-app="studentApp" 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>
```

OUTPUT :

Student Details

Student Name: SRINIVAS CGPA: 9.6 Add Student

Total Students: 1

- NISHYANTH KRISHNA - CGPA: 9.8

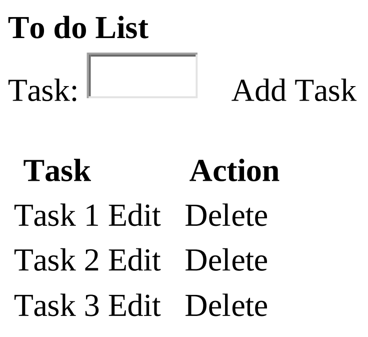
Program 6

Aim: 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>
```

OUTPUT:



To do List

Task: Add Task

Task	Action
Task 1 Edit	Delete
Task 2 Edit	Delete
Task 3 Edit	Delete

Program 7

Aim: Write an Angular JS 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>
 <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>
```

OUTPUT:

User Management

Name: [ ] Age: [ ] Add User

Name	**Age**	**Action**
Ram	25	Edit Delete
Sam	30	Edit Delete

Program 8

AIM: DevelopAngularJS program to create a login form, with validation for the username and password fields.

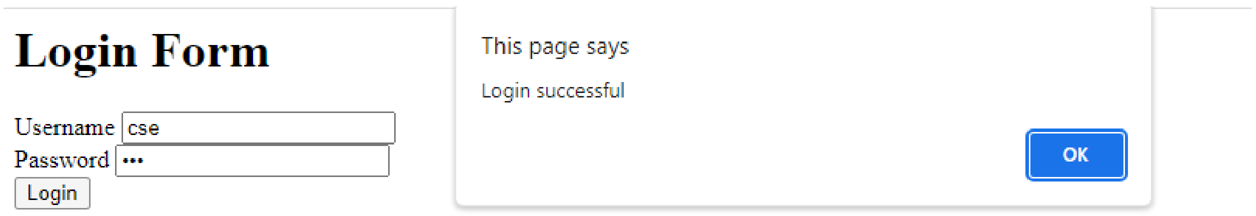
```
<html ng-app="loginApp">
 <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js"></script>
<body ng-controller="loginController">
 <h1>Login Form</h1>
 <!-- Form for login with validation -->
 <form ng-submit="login()">
 Username
 <input type="text" ng-model="username"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

OUTPUT:



Login Form

Username cse
Password •••
Login

This page says
Login successful
OK

Program 9

AIM: 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>

<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-app="employeeApp" 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>
```

OUTPUT:

```
    Employee List
Search by Name: [          ] Search by Salary:

  • Ram - Salary: Rs50000
  • abi - Salary: Rs60000
  • sam - Salary: Rs75000
  • raj - Salary: Rs55000
```

Program 10

AIM: 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>
```

OUTPUT:

Item Collection

Add New Item: [   ] Add Item

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

Total Items: 3

Program 11

AIM: 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>
```

OUTPUT:

Student Names

Original Names:

- Raj
- Ram
- Sam

Names in Uppercase:

- RAJ
- RAM
- SAM

Program 12

AIM : 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>
```

OUTPUT:

```
Date Display

Default Format: Feb 1, 2024

Custom Format (yyyy-MM-dd): 2024-02-01

Short Date: 2/1/24

Full Date: Thursday, February 1, 2024
```