JSS MAHAVIDYAPEETA

JSS ACADEMY OF TECHNICAL EDUCATION, BENGALURU

DEPARTMENT OF
COMPUTER SCIENCE AND ENGINEERING

Laboratory Manual

"Angular JS"

21CSL581

Introduction

AngularJS is a structural framework for dynamic web apps. It lets you use HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and succinctly. AngularJS's data binding and dependency injection eliminate much of the code you would otherwise have to write. And it all happens within the browser, making it an ideal partner with any server technology.

AngularJS is what HTML would have been, had it been designed for applications. HTML is a great declarative language for static documents. It does not contain much in the way of creating applications, and as a result building web applications is an exercise in *what do I have to do to trick the browser into doing what I want?*

The impedance mismatch between dynamic applications and static documents is often solved with:

- **a library** - a collection of functions which are useful when writing web apps. Your code is in charge and it calls into the library when it sees fit. E.g., jQuery.
- **frameworks** - a particular implementation of a web application, where your code fills in the details. The framework is in charge and it calls into your code when it needs something app specific. E.g., durandal, ember, etc.

AngularJS takes another approach. It attempts to minimize the impedance mismatch between document centric HTML and what an application needs by creating new HTML constructs. AngularJS teaches the browser new syntax through a construct we call *directives*. Examples include:

- Data binding, as in {{}}.
- DOM control structures for repeating, showing and hiding DOM fragments.
- Support for forms and form validation.
- Attaching new behavior to DOM elements, such as DOM event handling.
- Grouping of HTML into reusable components.

A complete client-side solution

AngularJS is not a single piece in the overall puzzle of building the client-side of a web application. It handles all of the DOM and AJAX glue code you once wrote by hand and puts it in a well-defined structure. This makes AngularJS opinionated about how a CRUD (Create, Read, Update, Delete) application should be built. But while it is opinionated, it also tries to make sure that its opinion is just a starting point you can easily change. AngularJS comes with the following out-of-the-box:

- Everything you need to build a CRUD app in a cohesive set: Data-binding, basic templating directives, form validation, routing, deep-linking, reusable components and dependency injection.
- Testability story: Unit-testing, end-to-end testing, mocks and test harnesses.
- Seed application with directory layout and test scripts as a starting point.

AngularJS's sweet spot

AngularJS simplifies application development by presenting a higher level of abstraction to the developer. Like any abstraction, it comes at a cost of flexibility. In other words, not every app is a good fit for AngularJS. AngularJS was built with the CRUD application in mind. Luckily CRUD applications represent the majority of web applications. To understand what AngularJS is good at, though, it helps to understand when an app is not a good fit for AngularJS.

Games and GUI editors are examples of applications with intensive and tricky DOM manipulation. These kinds of apps are different from CRUD apps, and as a result are probably not a good fit for AngularJS. In these cases it may be better to use a library with a lower level of abstraction, such as jQuery.

The Zen of AngularJS

AngularJS is built around the belief that declarative code is better than imperative when it comes to building UIs and wiring software components together, while imperative code is excellent for expressing business logic.

- It is a very good idea to decouple DOM manipulation from app logic. This dramatically improves the testability of the code.
- It is a really, *really* good idea to regard app testing as equal in importance to app writing. Testing difficulty is dramatically affected by the way the code is structured.
- It is an excellent idea to decouple the client side of an app from the server side. This allows development work to progress in parallel, and allows for reuse of both sides.
- It is very helpful indeed if the framework guides developers through the entire journey of building an app: From designing the UI, through writing the business logic, to testing.
- It is always good to make common tasks trivial and difficult tasks possible.

AngularJS frees you from the following pains:

- **Registering callbacks:** Registering callbacks clutters your code, making it hard to see the forest for the trees. Removing common boilerplate code such as callbacks is a good thing. It vastly reduces the amount of JavaScript coding *you* have to do, and it makes it easier to see what your application does.
- **Manipulating HTML DOM programmatically:** Manipulating HTML DOM is a cornerstone of AJAX applications, but it's cumbersome and error-prone. By declaratively describing how the UI should change as your application state changes, you are freed from low-level DOM manipulation tasks. Most applications written with AngularJS never have to programmatically manipulate the DOM, although you can if you want to.
- **Marshaling data to and from the UI:** CRUD operations make up the majority of AJAX applications' tasks. The flow of marshaling data from the server to an internal object to an HTML form, allowing users to modify the form, validating the form, displaying validation errors, returning to an internal model, and then back to the server, creates a lot of boilerplate code. AngularJS eliminates almost all of this boilerplate, leaving code that describes the overall flow of the application rather than all of the implementation details.
- **Writing tons of initialization code just to get started:** Typically you need to write a lot of plumbing just to get a basic "Hello World" AJAX app working. With AngularJS you can bootstrap your app easily using services, which are auto-injected into your application in a Guice-like dependency-injection style. This allows you to get started developing features quickly. As a bonus, you get full control over the initialization process in automated tests.

Installation Procedure

Before you commit to these installations, you should have basic working knowledge of:

- CSS
- HTML
- JavaScript

Additionally, it's helpful if you know TypeScript, but it's not mandatory. Here's what we recommend for the best outcome:

1. Installing Node.js

- Use this link to download the Node.js installer and install it on your system.
- Open the command prompt and type "npm-v" to check the version and installation.

2. Installing TypeScript

- Open this link and download the TypeScript installer, placing it in your device.
- Type the command "npm install -g typescript," running it on the command prompt.

3. Installing Angular CLI

a. Before you install Angular on your local Windows system, you must have these resources in place.

- Node.js. Angular needs an active LTS or maintenance LTS version. You should already have this installed back in Step 1. You can verify your Node.js installation by typing "$ node -v" in the command prompt.
- NPM Package Manager. Angular, Angular CLI, and all Angular applications rely on npm packages to accommodate most of their functions and features. Therefore, you will need an NPM packager manager to download and install the npm packages. Use the "npm-v" command to verify that you have the client installed.

- If you can't use a node version manager. If your system doesn't let you use an nvm, you can use this Node.js installer or this NodeSource installer.

b. Install Angular CLI with the npm package manager.

npm install -g @angular/cli

c. PowerShell script execution is disabled by default on Windows client computers. If you want to execute PowerShell scripts, something necessary for npm global binaries, set this execution policy, but read the message displayed after you run the command and follow the directions. Also, be sure you understand the implications of setting up an execution policy.

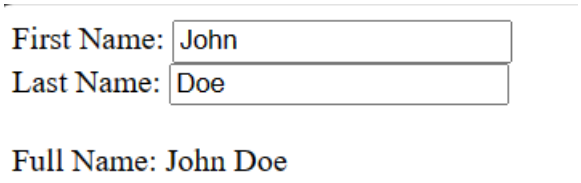
Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy RemoteSigned

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.

Execution Steps:
Step 1: Save the following program with .html extension
Step 2: Open the file using any browser

```
<!DOCTYPE html>
<html>
<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script
>
<body>

<div ng-app="myApp" ng-controller="myCtrl">

First Name: <input type="text" ng-model="firstName"><br>
Last Name: <input type="text" ng-model="lastName"><br>
<br>
Full Name: {{firstName + " " + lastName}}

</div>

<script>
var app = angular.module('myApp', []);
app.controller('myCtrl', function($scope) {
   $scope.firstName = "John";
   $scope.lastName = "Doe";
});
</script>

</body>
</html>
```

Output:



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.

Execution Steps:

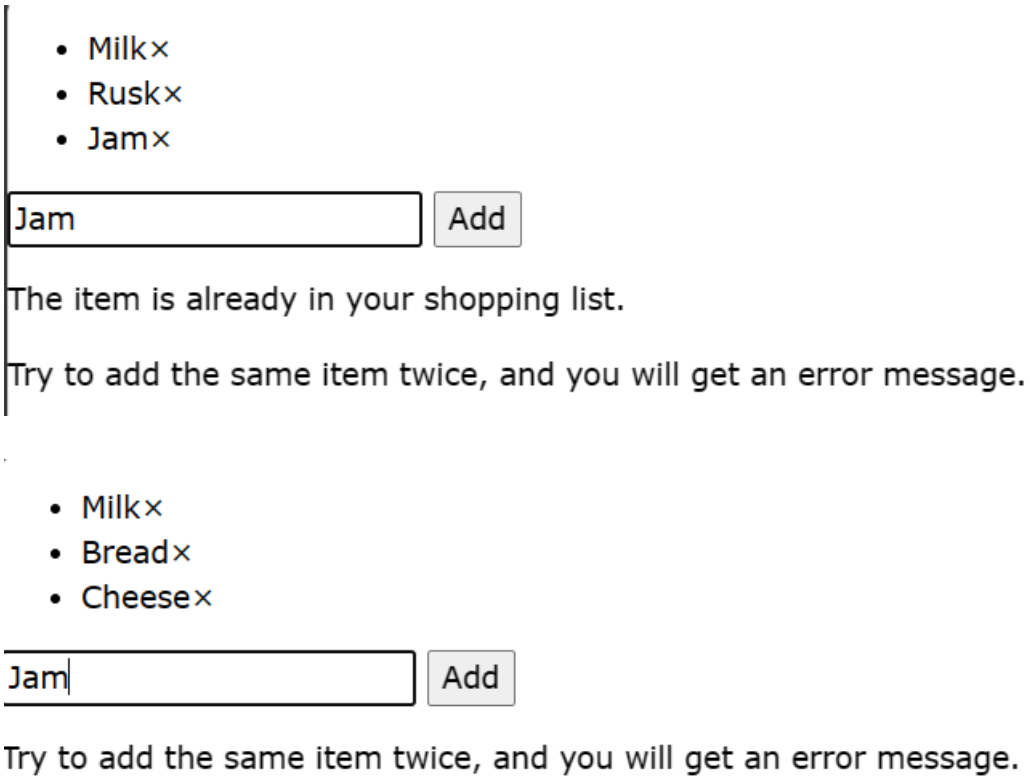
Step 1: Save the following program with .html extension

Step 2: Open the file using any browser

```
<!DOCTYPE html>
<html>

<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js">
</script>
<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
<body>

<script>


var app = angular.module("myShoppingList", []);
app.controller("myCtrl", function($scope) {
   $scope.products = ["Milk", "Bread", "Cheese"];
   $scope.addItem = function () {
      $scope.errortext = "";
      if (!$scope.addMe) {return;}
      if ($scope.products.indexOf($scope.addMe) == -1) {
         $scope.products.push($scope.addMe);
      } else {
         $scope.errortext = "The item is already in your shopping list.";
      }
   }
   $scope.removeItem = function (x) {
      $scope.errortext = "";
      $scope.products.splice(x, 1);
   }
});
</script>

<div ng-app="myShoppingList" ng-controller="myCtrl">
 <ul>
   <li ng-repeat="x in products">{{x}}<span ng-
click="removeItem($index)">×</span></li>
 </ul>
 <input ng-model="addMe">
 <button ng-click="addItem()">Add</button>
```

```
 <p>{{errortext}}</p>
</div>

<p>Try to add the same item twice, and you will get an error message.</p>

</body>
</html>
```

Output:



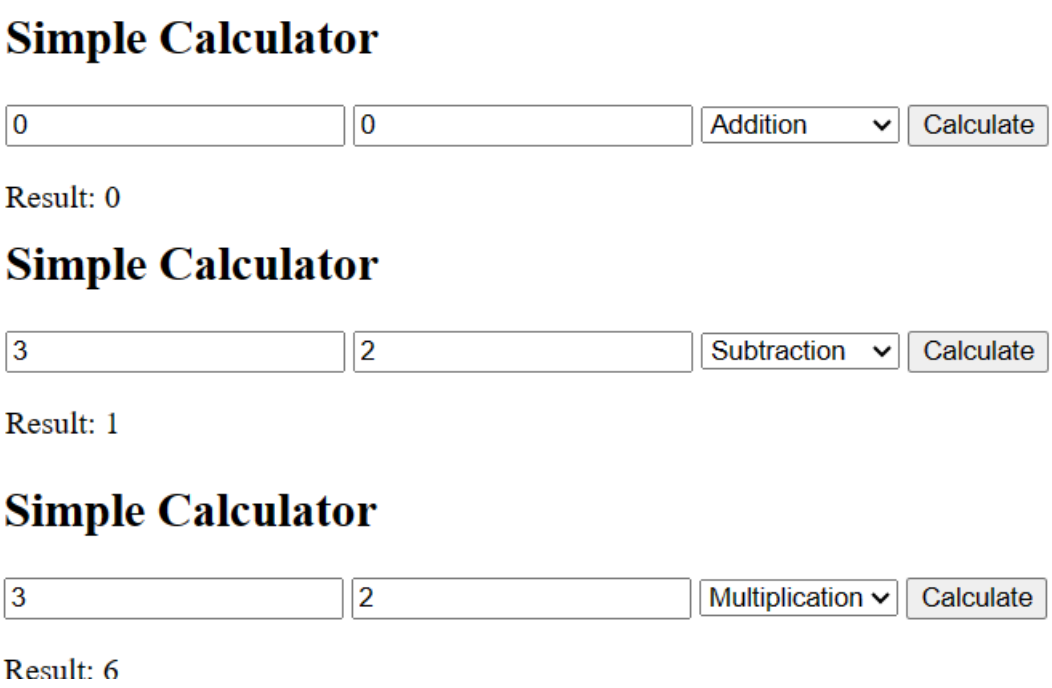
3. Develop a simple Angular JS calculator application that can perform basic mathematical operations (addition, subtraction, multiplication, division) based on user input.

Execution Steps:
Step 1: Save the following program with .html extension
Step 2: Open the file using any browser

```
<!DOCTYPE html>
<html ng-app="calculatorApp">
<head>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></sc
ript>
</head>
<body ng-controller="calculatorController">
  <h2>Simple Calculator</h2>
  <input type="number" ng-model="num1" placeholder="Enter number 1">
  <input type="number" ng-model="num2" placeholder="Enter number 2">
  <select ng-model="operator">
    <option value="+">Addition</option>
    <option value="-">Subtraction</option>
    <option value="*">Multiplication</option>
    <option value="/">Division</option>
  </select>
  <button ng-click="calculate()">Calculate</button>
  <p>Result: {{ result }}</p>

  <script>
    var app = angular.module('calculatorApp', []);
    app.controller('calculatorController', function($scope) {
      $scope.num1 = 0;
      $scope.num2 = 0;
      $scope.operator = '+';
      $scope.result = 0;

      $scope.calculate = function() {
        if ($scope.operator === '+') {
          $scope.result = $scope.num1 + $scope.num2;
        } else if ($scope.operator === '-') {
          $scope.result = $scope.num1 - $scope.num2;
        } else if ($scope.operator === '*') {
```

```
      $scope.result = $scope.num1 * $scope.num2;
    } else if ($scope.operator === '/') {
      if ($scope.num2 === 0) {
        $scope.result = 'Cannot divide by zero';
      } else {
        $scope.result = $scope.num1 / $scope.num2;
      }
    }
  };
});
</script>
</body>
</html>
```

Output:



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

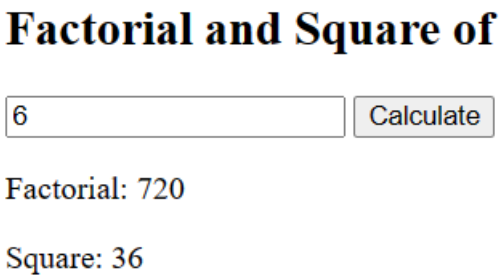
Execution Steps:

Step 1: Save the following program with .html extension

Step 2: Open the file using any browser

```
<!DOCTYPE html>
<html ng-app="Fact_SquareApp">
<head>
 <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script
>
</head>
<body ng-controller="FactController">
 <h2>Factorial and Square of a Number</h2>
 <input type="number" ng-model="num" placeholder="Enter number">

 <button ng-click="factcal()">Calculate</button>
 <p>Factorial: {{ fresult }}</p>
 <p>Square: {{ sresult }}</p>
 <script>
  var app = angular.module('Fact_SquareApp', []);
  app.controller('FactController', function($scope) {
    $scope.num = 0;
    $scope.fresult = 1;
    $scope.sresult = 1;

    $scope.factcal = function()
    {
     if ($scope.num === 0)
        {
      $scope.fresult = 1;
         $scope.sresult = 0
     }
        else
        {

          $scope.fresult = 1;
       for (var i = 2; i <= $scope.num; i++)
         $scope.fresult = $scope.fresult * i;

          $scope.sresult = $scope.num * $scope.num;
     }
    };
  });
 </script>
</body>
</html>
```

Output:


Factorial and Square of a Number
6
Calculate
Factorial: 720
Square: 36

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.

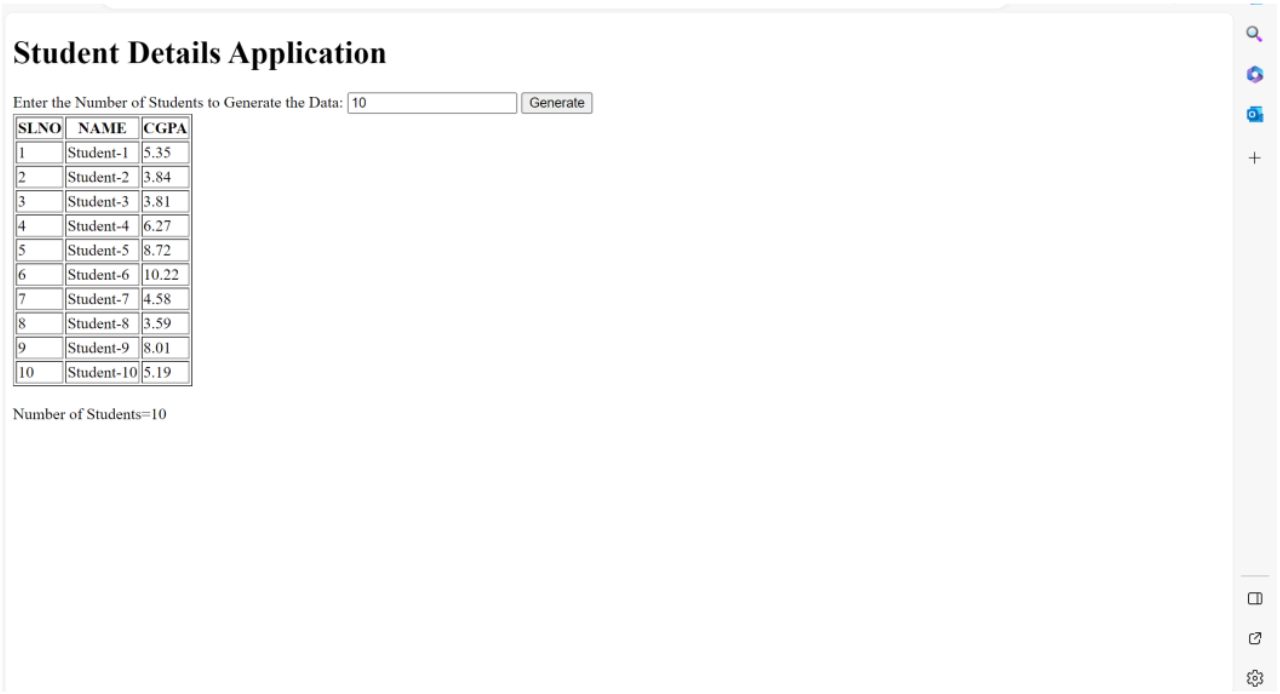
Execution Steps:

Step 1: Save the following program with .html extension

Step 2: Open the file using any browser

```
<!DOCTYPE html>
<html>
<title>Student Details Application</title>
<head>
<script type="text/javascript"
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js">
</script>
<script>
var app=angular.module("studDetailsApp",[]);
app.controller("studDetailsAppCntrl",function($scope){
$scope.studData=[]
$scope.generateData=function()
{
$scope.studData=[]
for(var i=1;i<=$scope.num;i++)
{
var stud={
"SLNO":i,
"NAME":'Student-'+i,
"CGPA":(Math.random()*10+1).toFixed(2)
}
$scope.studData.push(stud)
}
}
});
</script>
</head>
<body ng-app="studDetailsApp">
<h1>Student Details Application</h1>
<div ng-controller="studDetailsAppCntrl">
Enter the Number of Students to Generate the Data:
<input type="number" ng-model="num">
<button ng-click="generateData()">Generate</button>
<br/>
<table border="1" ng-show="studData.length>0">
<tr>
<th>SLNO</th>
<th>NAME</th>
<th>CGPA</th>
</tr>
<tr ng-repeat="student in studData">
<td>{{student.SLNO}}</td>
```

```
<td>{{student.NAME}}</td>
<td>{{student.CGPA}}</td>
</tr>
</table>
<br/>
Number of Students={{studData.length}}
</div>
</body>
</html>
```



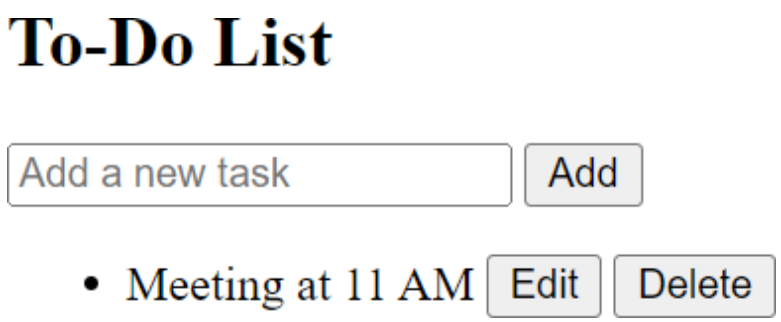
6. Develop an AngularJS program to create a simple to-do list application. Allow users to add, edit, and delete tasks.Note: The default values for tasks may be included in the program.

Execution Steps:
Step 1: Save the following program with .html extension
Step 2: Open the file using any browser

```
<!DOCTYPE html>
<html ng-app="ToDoApp">
<head>
  <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.mi
n.js"></script>
</head>
<body ng-controller="ToDoController">
  <h2>To-Do List</h2>

  <div>
    <input type="text" ng-model="newTask" placeholder="Add a new task">
    <button ng-click="addTask()">Add</button>
  </div>

  <ul>
    <li ng-repeat="task in tasks">
      {{ task.name }}
      <span ng-show="task.editing">
        <input type="text" ng-model="task.name" ng-blur="saveTask(task)">
      </span>
      <span ng-show="!task.editing">
        <button ng-click="editTask(task)">Edit</button>
        <button ng-click="removeTask(task)">Delete</button>
      </span>
    </li>
  </ul>

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

    app.controller('ToDoController', function($scope) {
      $scope.tasks = [];

      $scope.addTask = function() {
        if ($scope.newTask) {
          $scope.tasks.push({ name: $scope.newTask, editing: false });
          $scope.newTask = '';
        }
      };
```

```
    $scope.editTask = function(task) {
      task.editing = true;
    };

    $scope.saveTask = function(task) {
      task.editing = false;
    };

    $scope.removeTask = function(task) {
      const index = $scope.tasks.indexOf(task);
      if (index !== -1) {
        $scope.tasks.splice(index, 1);
      }
    };
  });
  </script>
</body>
</html>
```

Output:



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

Execution Steps:

Step 1: Create a new file with **.js** extension

Step 2: run the file in the terminal using the command **node filename.js**

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title></title>
  <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.4.5/angular.min.js"></script>
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.0/jquery.min.js"></script>
</head>
<body>

  <div ng-app="mainApp" data-ng-controller="CRUDController">

    <table>
      <tr>
        <td>EmpId: </td>
        <td>
          <span>{{ EmpModel.Id }}</span></td>
      </tr>
      <tr>
        <td>Name:</td>
        <td>
          <input type="text" data-ng-model="EmpModel.Name" /></td>
      </tr>
      <tr>
        <td>Salary:</td>
        <td>
          <input type="number" data-ng-model="EmpModel.Salary" /></td>
      </tr>
      <tr>
        <td>
          <input type="button" value="Save Data" data-ng-click="AddData()" /></td>
        <td>
          <input type="button" value="Update Data" data-ng-click="UpdateData()" /></td>
      </tr>
    </table>

    <table border="1" style="width: 300px">
      <thead>
        <th>Id</th>
        <th>Name</th>
        <th>Salary</th>
      </thead>
      <tr data-ng-repeat="Emp in EmpList" data-ng-click="BindSelectedData(Emp)">
        <td>{{ Emp.Id }}</td>
        <td>{{ Emp.Name }}</td>
        <td>{{ Emp.Salary }}</td>
```

```
        <td>
          <input type="button" value="Delete" data-ng-click="DeleteData(Emp)" /></td>
      </tr>
    </table>
  </div>

  <script type="text/javascript">
    var app = angular.module("mainApp", []);
    app.controller('CRUDController', function ($scope) {

      $scope.EmpModel = {
        Id: 1,
        Salary: 0,
        Name: '',
      };

      $scope.EmpList = [];
      $scope.AddData = function () {
        var _emp = {
          Id: $scope.EmpList.length + 1,
          Name: $scope.EmpModel.Name,
          Salary: $scope.EmpModel.Salary
        };
        $scope.EmpList.push(_emp);
        ClearModel();
      }

      $scope.DeleteData = function (emp) {
        var _index = $scope.EmpList.indexOf(emp);
        $scope.EmpList.splice(_index, 1);
ClearModel();
      }

      $scope.BindSelectedData = function (emp) {
        $scope.EmpModel.Id = emp.Id;
        $scope.EmpModel.Name = emp.Name;
        $scope.EmpModel.Salary = emp.Salary;
      }

      $scope.UpdateData = function () {
        $.grep($scope.EmpList, function (e) {
          if (e.Id == $scope.EmpModel.Id) {
            e.Name = $scope.EmpModel.Name;
            e.Salary = $scope.EmpModel.Salary;
          }
        });
        ClearModel();
      }

      function ClearModel() {
        $scope.EmpModel.Id = $scope.EmpList.length + 1;
        $scope.EmpModel.Name = '';
        $scope.EmpModel.Salary = 0;
      }
    });
  </script>
```

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

Output:

EmpId: 3

Name:

Salary: 0

Save Data | Update Data

Id	Name	Salary	
1	Supriya N	0	Delete
2	Shilpashree	0	Delete

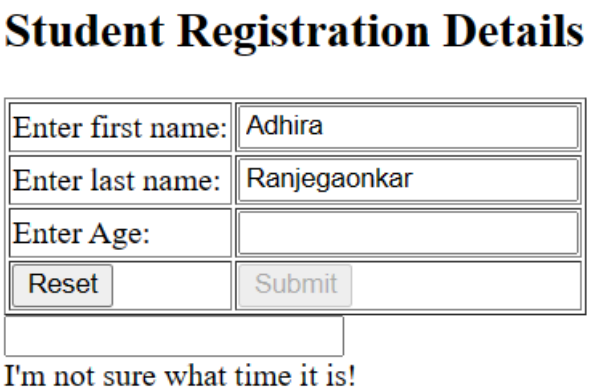
8. Develop AngularJS program to create a login form, with validation for the username and password fields.

```
<!DOCTYPE html>
<html ng-app="mainApp">
<head>
 <script
   src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></s
   cript>
</head>
<body ng-controller="studentController">
 <h2>Student Registration Details</h2>
 <div>
  <form name="studentForm" novalidate>
   <table border="1">
    <tr>
     <td>Enter first name:</td>
     <td>
      <input name="firstName" type="text" ng-model="firstName" ng-
   pattern="/^[a-zA-Z]*$/" required>
      <span style="color:red" ng-show="studentForm.firstName.$dirty &&
   studentForm.firstName.$invalid">
       <span ng-show="studentForm.firstName.$error.required">First Name is
   required.</span>
       <span style="color:red" ng-
   show="studentForm.firstName.$error.pattern">*Invalid First name.</span>
      </span>
     </td>
    </tr>
    <tr>
     <td>Enter last name: </td>
     <td>
      <input name="lastName" type="text" ng-model="lastName" ng-
   pattern="/^[a-zA-Z]*$/" required>
      <span style="color:red" ng-show="studentForm.lastName.$dirty &&
   studentForm.lastName.$invalid">
       <span ng-show="studentForm.lastName.$error.required">Last Name is
   required.</span>
       <span style="color:red" ng-
   show="studentForm.lastName.$error.pattern">*Invalid Last name.</span>
      </span>
     </td>
    </tr>
```

```
    <tr>
     <td>Enter Age:</td>
     <td>
      <input type="text" ng-model="age" name="age" ng-pattern="/^(?:1[8-9]|[2-
  5][0-9]|50)$/" required/>
      <span style="color:red" ng-show="studentForm.age.$dirty &&
  studentForm.age.$invalid">
       <span ng-show="studentForm.age.$error.required">Age is required.</span>
       <span style="color:red" ng-
  show="studentForm.age.$error.pattern">*Invalid Age. Valid 18-50</span>
      </span>
     </td>
    </tr>
    <tr>
     <td>
      <button ng-click="reset()">Reset</button>
     </td>
     <td>
      <button ng-disabled="studentForm.$invalid" ng-
  click="submit()">Submit</button>
     </td>
    </tr>
   </table>
  </form>
</div>
<div>
 <input type="number" ng-model="value"><br>
 <span>{{ value | greet }}</span>
</div>
<script>
 var mainApp = angular.module("mainApp", []);
 mainApp.controller('studentController', function ($scope) {
  $scope.reset = function () {
   $scope.firstName = 'Adhira';
   $scope.lastName = 'Ranjegaonkar';
   $scope.age = '';
  }
  $scope.reset();
 });
```

```
  mainApp.filter('greet', function() {
    return function(input) {
      if (input < 12) {
        return 'Good Morning';
      } else if (input >= 12 && input <= 17) {
        return 'Good Afternoon';
      } else if (input > 17 && input <= 24) {
        return 'Good Evening';
      } else {
        return "I'm not sure what time it is!";
      }
    };
  });
 </script>
</body>
</html>
```

Output:

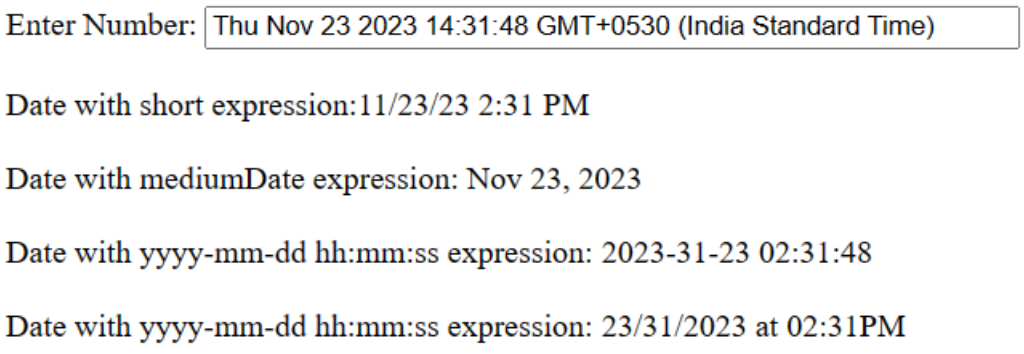


12. Create an AngularJS application that displays the date by using date filter parameters

```
<!DOCTYPE html>
<html>

<head>

<title>

    AngularJs Date filter Example

</title>

<script
src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.m
in.js"></script>

<script>

var app = angular.module("AngulardateApp", []);

app.controller("datectrl", function ($scope) {

$scope.sampledate = new Date();

});

</script>

</head>

<body ng-app="AngulardateApp">

<div ng-controller="datectrl">

Enter Number: <input type="text" ng-model="sampledate"
style="width:400px" /><br /><br />

Date with short expression:{{sampledate | date:"short" }}<br /><br
/>

Date with mediumDate expression: {{sampledate | date :
"mediumDate"}} <br /><br />

Date with yyyy-mm-dd hh:mm:ss expression: {{sampledate | date :
"yyyy-mm-dd hh:mm:ss" : 0}} <br /><br />
```

```
Date with yyyy-mm-dd hh:mm:ss expression: {{sampledate | date :
"dd/mm/yyyy 'at' hh:mma" : 0}}

</div>

</body>

</html>
```

Output :



VIVA QUESTIONS

Q #1) What do you understand by AngularJS?

Answer: AngularJS is a JavaScript framework that is used for making rich and extensible web applications.

It runs on plain JavaScript and HTML, hence you don't need any other dependencies to make it work. AngularJS is perfect for Single Page Applications (SPA). It is basically used for binding JavaScript objects with HTML UI elements.

Q #2) Define the features of AngularJS.

Answer: The features include:

- The Template (View)
- The Scope (Model)
- The Controller (Controller)
- Services
- Filters
- Directives

Q #3) Define Data Binding.

Answer: Data binding is an automatic attunement of data between the view and model components.

Q #4) Distinguish between AngularJS and JavaScript expressions.

Answer: There are several differences between AngularJS and JavaScript expressions:

- We can write AngularJS expressions in HTML, but we cannot write JavaScript expressions in HTML.
- We cannot use conditional iterations, loops, and exceptions in AngularJS, but we can use all of these conditional properties in JavaScript expressions.
- Filters are supported in AngularJS whereas filters are not supported in JavaScript.

Q #5) Write all the steps to configure an **Angular App(ng-app).**

Answer: To set up an Angular App we must follow certain steps as mentioned below:

- angular.module will be created at first.
- A controller will be assigned to the module.
- The module will be linked with the HTML template(i.e. UI or View) with an angular app(ng-app).
- The HTML template will be linked with the controller(i.e JS) with an ng-controller directive.

Q #6) What are the Angular Modules?

Answer: The angular modules collectively define an angular application where we can write the angular code. Modules contain the different parts of an angular application. A module is created by

angular.module function in angular.

Q #7) What are the directive scopes in AngularJS?

Answer: Three directive scopes are available in AngularJS.

They are:

- **Parent scope:** Whatever change you make in your directive that comes from the parent scope, will also reflect in the parent scope, and it is also a default scope.
- **Child scope:** It is a nested scope that inherits a property from the parent scope. Also, if any properties and function on the scope are not connected with the parent scope directive, then a new child scope directive is created.
- **Isolated scope:** It is reusable and is used when we build a self-contained directive. It is only used for private and internal use which means that it does not contain any properties of the parent scope.

Q #8) How can we share the data between controllers in AngularJS?

Answer: First, we have to create a service. Service is used to share the data between controllers in AngularJS in a very lucid, easy and fastest way. We use events, $parent, next sibling, and controller by using a $rootScope.

Q #9) What is the digest cycle in AngularJs?

Answer: It is a part of the process of data binding in AngularJS. It compares the old and new versions of the scope model value in each digest cycle.

The digest cycle is triggered automatically. We can also enhance the usability by using $apply () if we want to trigger the digest cycle manually.

Q #10) Explain the differences between one-way binding and two-way binding.

Answer: One-way binding is used to bind the data from the model to view without updating the HTML template or view automatically.

Thus, in order to update the HTML template, we need to write a custom code that will update the view every time whenever a data is binded from model to view.

Whereas, two-way binding is used to bind the data from the model to view and vice versa(i.e view to model) by automatically updating the HTML template without writing any custom code.

Q #11) Difference between sessionStorage, cookies, and localStorage.

Answer: The differences are as follows:

- **SessionStorage –** The data is stored for a particular session. The data will be lost whenever the browser tab will be closed or after some particular session. The maximum size stored can be up to 5MB.
- **LocalStorage –** The data is stored with no expiration date. The data can only be cleared by JavaScript or by clearing the browser cache. The storage limit is maximum than the sessionStorage and cookie.
- **Cookies –** It stores the data that has to be sent back to the server with some requests. The cookie's expiration varies on the type and duration set from either the server-side or client-side. The maximum size stored can be less than 4KB.

Q #12) What is the role of $routeProvider in AngularJS?

Answer: It is the $routeProvider that helps in navigating between different pages/links without separately loading the page/link whenever a user clicks on a link.

ngRoute config() method is used to configure the routeProvider.

Q #13) What is the difference between $scope and scope?

Answer: In AngularJS, $scope is used to achieve dependency injection and scope is used for linking between View (i.e HTML) and Controller (i.e JS).

Q #14) How are AngularJS prefixes $ and $$ used?

Answer: $$ variable in AngularJS is used as a private variable, as it is used to prevent accidental code collision with the user code.

Whereas, $ prefix can be used to denote angular core functionalities (like a variable, parameter, property or method).

Q #15) Where can we implement the DOM manipulation in AngularJS?

Answer: Manipulation of DOM is in directives and apart from this it should not exist in the controller's services or anywhere else.

Q #16) How can we show that a scope variable should have one-time binding only?

Answer: To show one-time binding we have to use "**::**" in front of the scope.

Q #17) What is SPA (Single Page Application) in AngularJS?

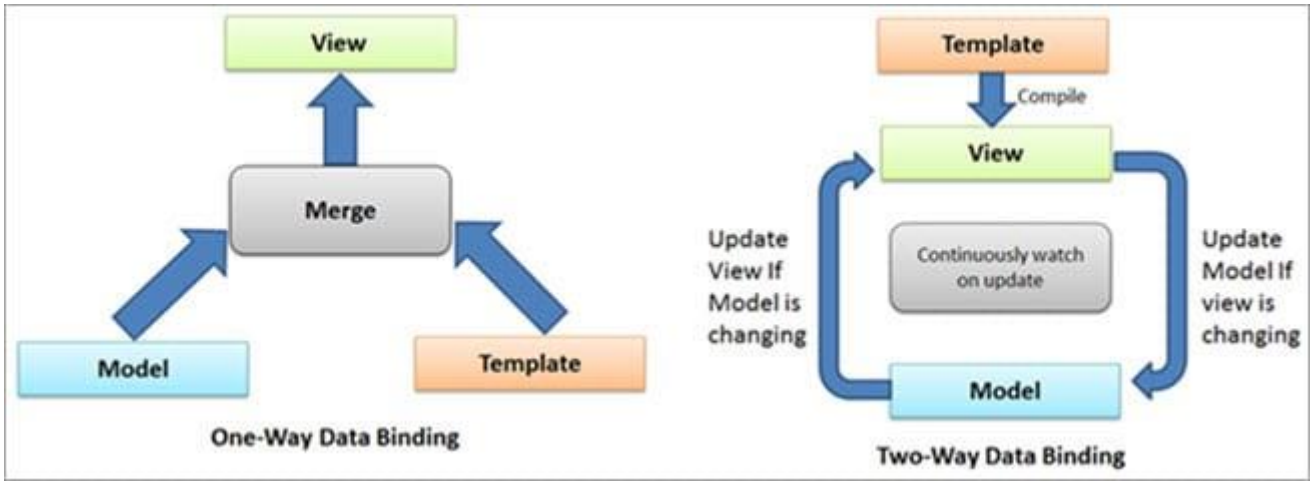
Answer: It is a web application that loads a single HTML page and dynamically updates the page as the user connects with the app.

By using AJAX and HTML a fluid and responsive web app can be created by SPA without invariant page reloads. Through this, we can make responsive UI with no page flicker.

Q #18) How many types of data bindings are there in AngularJS?

Answer: AngularJS supports both one way and two-way binding.

In one way binding if we change the data model, then there will be no dynamic change that you will see in view but in two way binding, there will be a dynamic change whenever a change will be made in the data model.



Q #19) What are the binding directives in AngularJs?

Answer: The binding directives include:

- ng-bind
- ng-bind-html
- ng-bind-template
- ng-non-bindable
- ng-model

Q #20) Explain ng-bind and ng-bind-html directives.

Answer:

ng-bind: It is a directive that replaces the content of the HTML element with the value of the assigned variable or expression.

The content of the HTML element will change by changing the value of the variable or expression.

It is like ({{expression}}) and the syntax for this is,

<ANY ELEMENT ng-bind="expression"> </ANY ELEMENT>

ng-bind-html: It is a directive that binds the content to the HTML element(view) in a secure way. $sanitize service is used to sanitize the content to bind into an HTML element. To do this 'angular-sanitize.js' must be included in our application.

Syntax to write this,

<ANY ELEMENT ng-bind-html=" expression "> </ANY ELEMENT>

Q #21) Explain ng-bind-template and ng-non-bindable.

Answer:

ng-bind-template: It replaces the text content of the element by interpolation of the template. It can contain multiple double curly markups.

<ANY ELEMENT ng-bind-template= " {{expression1}} {{expression2}} … {{expression n}} ">
</ANY ELEMENT>

Ng-non-bindable: It specifies AngularJS to not compile the content of this HTML element and its child nodes.

<ANY ELEMENT ng-non-bindable > </ANY ELEMENT>

Q #22) Explain the ng-model directive in AngularJs.

Answer: This can be a leap hop with the custom HTML input form control(like input, textarea and select) to the application data. It provides form validation behavior with two-way binding.

<input ng-bind="expression"/>

Q #23) Define the Factory method in AngularJS.

Answer: It is quite similar to service, factories implement a module pattern in which we use a factory method to generate an object which is used for building models.

In a factory, a method object is returned at the end by creating a new object and adding functions as properties.

Syntax:

module.factory(‘factoryName', function);

Q #24) What is ng-repeat directive in AngularJS?

Answer: It renders or iterates over a collection of items and creates DOM elements. It regularly monitors the source of data to re-render a template in response to a change.

Syntax:

<table class="table table-bordered">

<tr ng-repeat="student stuDetails">

<td>{{stu.name}} </td>

```
        <td> {{stu. grade}} </td>

    </tr>

</table>
```

Q #25) What is a controller in AngularJS?

Answer: A controller is a JavaScript function that is bound to the specified scope. Angular instantiates the new controller object and injects the new scope as a dependency.

A controller can be used to set up the initial state of the scope object and to add behavior to the object. A controller cannot be used to share code or state across controllers, but instead of that Angular service can be used.

```
<Any ng-Controller=" expression">

</Any>

<div ng-app="mainApp" ng-controller="SimpleController">

</div>
```

Q #26) What are filters in AngularJS?

Answer: The main work of filters is to modify the data, so that it can be merged into an expression or directive by using a pipe character (it is used for applying filters in an angular symbol of a pipe which is (|) or this is the symbol).

A filter formats the value of an expression for a display to the user. They can be used in view templates, controllers, or services, and we can easily create our own filter as well. A filter is a module provided by AngularJS. There are nine components of a filter which are provided by it.

Examples: currency, date, filter, JSON, limitTo, etc.

Q #27) What is ng-App directive in AngularJS?

Answer: It is used to define the AngularJs Application. It appoints the root element of the application and it is kept near the <body> or <html> tag.

We can define any number of ng-app directives inside the HTML document, but only one AngularJS application can be bootstrapped automatically (auto-bootstrapped) and the other applications need to be bootstrapped manually.

Example:

```
<div ng-app="">

<p>My first expression: {{157 + 122}} </p>

</div>
```

Q #28) What is ng-switch in AngularJS?

Answer: It is used to conditionally exchange the structure of DOM on a template that is based on a scope-based expression.

This directive lets you show or hide the HTML element depending on the expression.

```
<element ng-switch="expression">

<element ng-switch-when="value"></element>
```

Q #29) What is the use of a double-click event in AngularJs?

Answer: It allows you to specify the custom behavior on a double click event of the mouse on a web page. We can use it (ng-dblclick) as an attribute of the HTML element like,

```
<ANY_HTML_ELEMENT ng-dblclick="{expression}">

...

</ANY_HTML_ELEMENT>
```

Q #30) What are ng-include and ng-click directives in AngularJs?

Answer:

ng-include helps to include different files on the main page. The ng-include directive includes HTML from an external file.

The included content will be included as child nodes of the specified element. The value of the ng-include attribute can also be an expression, returning a filename.

By default, the included file must be located on the same domain as the document.

ng-click can be used in scenarios like when you click on a button or when you want to do any operation. It tells AngularJS what to do when an HTML element is clicked.

Example:

<button ng-click="count = count + 1" ng-init="count=0">OK</button>

The above code will increase the count variable by one whenever the button is clicked.

Q #31) What is a representational state transfer(REST) in AngularJs?

Answer: REST is an API style that operates over the HTTP request.

The requested URL identifies the data to be operated on, and the HTTP method identifies the operation that is to be performed. REST is a style of API rather than a formal specification, and there is a lot of debate and disagreement about what is and isn't RESTful, which is a term used to indicate an API that follows the REST style.

AngularJS is flexible about how RESTful web services are consumed.

Q #32) What are the AngularJs Global API?

Answer: It is a combination of global JavaScript function which is used to perform tasks like comparing objects, iterating objects and converting data.

There are some common API functions like:

- **angular. lowercase:** It converts a string to lowercase string.
- **angular. uppercase:** It converts a string to uppercase string.
- **angular. isString:** It will return true if the current reference is a string.
- **angular. isNumber:** It will return true if the current reference is a number.

Q #33) What is a provider method in AngularJs?

Answer: A provider is an object which creates a service object by allowing to take more control.

$get() method is used in the provider which returns the service object. The service name and the factory function are the arguments that are passed into the provider method. AngularJS uses $provide to register new providers.

Syntax:

serviceApp.provider("logService", function ())

Q #34) What is Event Handling?

Answer: Event handling in AngularJs is very useful when you want to create advance AngularJs applications.

We need to handle DOM events like mouse clicks, moves, keyboard presses, change events and so on. AngularJs has some listener directives like ng-click, ng-dbl-click, ng-mousedown, ng-keydown, ng-keyup etc.

Q #35) What is AngularJs DOM?

Answer: AngularJs have some directives which are used to encapsulate AngularJs application data to a disabled attribute of the HTML elements.

Example: ng-disabled directive encapsulates the application data to the disabled attributes of the HTML DOM element.

```
<div ng-app="" ng-init="mySwitch=true">

<p>

<button ng-disabled="mySwitch">Click Me!</button>

</p>

<p>

<input type="checkbox" ng-model="mySwitch"/>Button

</p>

<p>

{{ mySwitch }}

</p>

</div>
```

Q #36) What are the attributes that can be used during the creation of a new AngularJs

directives?

Answer: There are several attributes that can be used during a new directive creation.

They include:

1. **Template:** It describes an inline template as a string.
2. **Template URL:** This attribute specifies the AngularJs HTML compiler to replace the custom directive inside a template with the HTML content located inside a separate file.
3. **Replace:** It replaces the current element if the condition is true if false append this directive to the current element.
4. **Transclude:** It allows you to move the original children of a directive to a location inside the new template.
5. **Scope:** It creates a new scope for this directive rather than inheriting the parent scope.
6. **Controller:** It creates a controller which publishes an API for communicating across the directives.
7. **Require:** It requires another directive to be present to function the current directive efficiently.
8. **Link:** It modifies resulting in DOM element instances, adds event listeners, and set up data binding.
9. **Compile:** It modifies the DOM template for features across copies of a directive, as when used in other directives. Your compile function can also return link functions to modify the resulting element instances.

Q #37) Is Nested Controllers possible or not in AngularJs?

Answer: Yes, it is possible as Nested Controllers are well-defined in a classified manner while using a view.

Q 38) Is AngularJS well-suited with all browsers?

Answer: Yes, it is companionable with all browsers like Safari, Chrome, Mozilla, Opera, IE, etc. as well as mobile browsers.

Q 39) Define services in AngularJS.

Answer: AngularJS services are the singleton objects or functions which are used for carrying out definite tasks. It embraces some corporate ideas and these purposes can be called controllers, directive, filters and so on.

Q 40) Explain the advantages of AngularJS.

Answer: Advantages include:

- It supports MVC form.
- Organize two ways of data binding using AngularJS.
- It supports mutual client-server communication.

- It supports simulations.

Q #41) Difference between services and factory.

Answer: Factories are functions that return the object, while services are constructor functions of the object which is used by a new keyword.

Syntax:

Factory – module.factory(`factoryName`, function);

Service – module.service(`serviceName`, function);

Q #42) If both factory and service are equivalent, then when should I use them?

Answer: Factory provider is preferred using an object, whereas a service provider is preferred using with class.

Q #43) Difference between AngularJS and React.JS.

Answer: AngularJS is a TypeScript language-based JS framework released in October 2010 by Google. It is a completely free framework and open source that is used in SPA projects (i.e. Single Page Application projects).

React.JS is a javascript library developed by Facebook in March 2013 for building UI. React components can be used on several pages but not as a SPA (i.e. Single Page Application).

Q #44) Difference between ng-bind and ng-model directive.

Answer: ng-bind directive has one-way data bindings, data flows only from object to UI, not vice versa (i.e. $scope>>view) and ng-model directive has two-way data bindings, data flows between UI to object and vice versa(i.e. $scope>>view and view>>$scope).

Q #45) What is the difference between AJAX and AngularJS?

Answer: AJAX stands for Asynchronous JavaScript and XML which is used for sending and getting responses from the server without loading the page.

Whereas, AngularJS is a typescript language-based JavaScript framework following the MVC pattern.

Q #46) Define ng-if, ng-show and ng-hide.

Answer: ng-if directive is used as if clause which removes the HTML element if the expression becomes false.

Syntax

<element ng-if=”expression”></element>

ng-show directive is used to show the HTML element if the expression becomes true. And if the expression becomes false then the HTML element will be hidden.

Syntax

<element ng-show=”expression”></element>

ng-hide directive is used to hide the HTML element if the expression becomes false.

Syntax

<element ng-hide=”expression”></element>

Both ng-show and ng-hide uses the display property method.

Q #47) What is the difference between ngRoute and ui-router?

Answer: ngRoute is a module developed by angularJS team which was a part of the core angularJS framework. Whereas ui-router was developed by a third-party community to overcome the problems of ngRoute.

ngRoute is a location or URL based routing, and ui-router is a state-based routing which allows nested views.

Further Reading => Most Popular Full Stack Developer Interview Questions

Q #48) How to set, get and clear cookies in AngularJs?

Answer: AngularJS has a module called ngCookies, so before injecting ngCookies angular-cookies.js should be included in the application.

- **Set Cookies** – Put method is used to set cookies in a key-value format.

$cookies.put(“username”, $scope.username);

- **Get Cookies –** Get method is used to get cookies.

$cookies.get(‘username’);

- **Clear Cookies –** Remove method is used to remove cookies.

$cookies.remove(‘username’);