**MODULE 6**

**AIM:**

To Develop a Single Page Application Using AngularJS

**OBJECTIVE:**

Create single page applicaiton in which all the code (JS, HTML, CSS) is loaded when application loads for the first time. Loading of the dynamic contents and the navigation between pages is done without refreshing the page.

**THEORY**

Traditionally, applications were Multi-Page Application (MPA) where with every click a new page would be loaded from the server. This was not only time consuming but also increased the server load and made the website slower. AngularJS is a JavaScript-based front-end web framework based on bidirectional UI data binding and is used to design Single Page Applications. Single Page Applications are web applications that load a single HTML page and only a part of the page instead of the entire page gets updated with every click of the mouse. The page does not reload or transfer control to another page during the process. This ensures high performance and loading pages faster. Most modern applications use the concept of SPA. In the SPA, the whole data is sent to the client from the server at the beginning. As the client clicks certain parts on the webpage, only the required part of the information is fetched from the server and the page is rewritten dynamically. This results in a lesser load on the server and is cost-efficient. SPAs use AJAX and HTML5 to create a fluid and responsive Web applications and most of the work happens on the client-side. Popular applications such as Facebook, Gmail, Twitter, Google Drive, Netflix, and many more are examples of SPA.

SPAs are good when the volume of data is small and the website that needs a dynamic platform. It is also a good option for mobile applications. But businesses that depend largely on search engine optimizations such as e-commerce applications must avoid single-page applications and opt for MPAs.

**ALGORITHM**

**Step 1:**

**Create a Module**

AngularJS follows MVC architecture, hence every AngularJS application contains a module comprising of controllers, services, etc.

var app = angular.module('myApp', []);

**Step 2:**

**Define a Simple Controller**

app.controller('FirstController', function($scope) {

$scope.message = 'Hello from FirstController';

});

**Step 3:**

**Include AngularJS script in HTML code**

Specify the module (created in step 1) in **ng-app** attribute and the controller (defined in step 2) in **the ng-controller** attribute.

<!doctype html>

<html ng-app="myApp">

<head>

<script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.4.7/angular.min.js"></script>

</head>

<body ng-controller="FirstController">

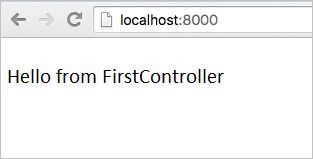
<h1>{{message}}</h1>

<script src="app.js"></script>

</body>

</html>

Once the code is run on localhost, the **output** will be as shown below.



It is now confirmed that our module and controller are set up, and AngularJS is working properly.

**Step 4:**

**Use AngularJS’s routing capabilities to add different views to our SPA**

Include **angular-route** script after the main angular script.

<script src="https://cdnjs. cloudflare.com/ajax/libs/angular.js /1.4.7 /angular.min.js"> </script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.4.7/angular-route.min.js"></script>

Use the **ngRoute** directive to enable routing.

var app = angular.module('myApp', ['ngRoute']);

**Step 5:**

**Create an HTML layout for the website**

Once an HTML layout for the website is created, use the**ng-view**directive to specify the place where the HTML of each page will be placed in our layout.

<!doctype html>

<html ng-app="myApp">

<head>

<script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.4.7/angular.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.4.7/angular-route.min.js"></script>

</head>

<body>

<div ng-view></div>

<script src="app.js"></script>

</body>

</html>

**Step 6:**

**Use $routeProvider service from ngRoute module to configure the navigation to different views**

It is necessary to specify a templateUrl and a controller for each route that we wish to add.

Exception handling has to be accommodated when a user tries to navigate to a route that doesn’t exist. For simplicity, we can write an “otherwise” function to redirect the user to the “/” route.

|  |
| --- |
| var app = angular.module('myApp', ['ngRoute']);    app.config(function($routeProvider) {  $routeProvider    .when('/', {  templateUrl : 'pages/first.html',  controller : 'FirstController'  })    .when('/blog', {  templateUrl : 'pages/second.html',  controller : 'SecondController'  })    .when('/about', {  templateUrl : 'pages/third.html',  controller : 'ThirdController'  })    .otherwise({redirectTo: '/'});  }); |

**Step 7:**

**Build controllers for every route specified in $routeProvider**

|  |
| --- |
| app.controller('FirstController', function($scope) {  $scope.message = 'Hello from FirstController';  });    app.controller('SecondController', function($scope) {  $scope.message = 'Hello from SecondController';  });    app.controller('ThirdController', function($scope) {  $scope.message = 'Hello from ThirdController';  }); |
|  | |

**Step 8:**

**Configure the pages**

**first.html**

<h1>First</h1>

<h3>{{message}}</h3>

**second.html**

<h1>Second</h1>

<h3>{{message}}</h3>

**third.html**

<h1>Third</h1>

<h3>{{message}}</h3>

**Step 9:**

**Add links to the HTML that will help in navigating to the configured pages**

<!doctype html>

<html ng-app="myApp">

<head>

<script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.4.7/angular.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.4.7/angular-route.min.js"></script>

</head>

<body>

<a href="#/">First</a>

<a href="#/second">Second</a>

<a href="#/third">Third</a>

<div ng-view></div>

<script src="app.js"></script>

</body>

</html>

**Step 10:**

**Include the HTML of routing pages to index.html file using script tag**

<!doctype html>

<html ng-app="myApp">

<head>

<script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.4.7/angular.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.4.7/angular-route.min.js"></script>

</head>

<body>

<script type="text/ng-template" id="pages/first.html">

<h1>First</h1>

<h3>{{message}}</h3>

</script>

<script type="text/ng-template" id="pages/second.html">

<h1>Second</h1>

<h3>{{message}}</h3>

</script>

<script type="text/ng-template" id="pages/third.html">

<h1>Third</h1>

<h3>{{message}}</h3>

</script>

<a href="#/">First</a>

<a href="#/second">Second</a>

<a href="#/third">Third</a>

<div ng-view></div>

<script src="app.js"></script>

</body>

</html>

(When Angular detects the templates defined by the ng-template directives, it will load its content to the template cache and will not perform Ajax request to get their content.)

**PROGRAM**

<!doctype html>

<html ng-app="myApp">

<head>

<script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.4.7/angular.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/angular.js/1.4.7/angular-route.min.js"></script>

</head>

<body>

<script type="text/ng-template" id="pages/first.html">

<h1>First</h1>

<h3>{{message}}</h3>

</script>

<script type="text/ng-template" id="pages/second.html">

<h1>Second</h1>

<h3>{{message}}</h3>

</script>

<script type="text/ng-template" id="pages/third.html">

<h1>Third</h1>

<h3>{{message}}</h3>

</script>

<a href="#/">First</a>

<a href="#/second">Second</a>

<a href="#/third">Third</a>

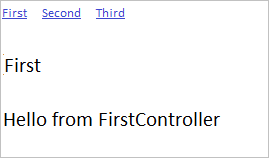
<div ng-view></div>

<script src="app.js"></script>

</body>

</html>

Once the HTML is run on localhost, the following page is displayed.



Observe that the hyperlinks **First, Second, Third** on the page are routers and when you click on them, navigation to the corresponding web pages occur, without refreshing.

