1) Angular JS = Used In Single Page Application (SPA).

2) It is very powerful JavaScript Library.

3) It extends HTML DOM Attributes and make responsive User Action. [<div class='head' ng-*=''>]

4) Angular JS is open source Web application Framework.

Features:-

5) Angular JS is powerful JavaScript development framework to Create RICH Internet Application (RIA).

6) Angular JS provides developer Option to write Client Side Application (Using JavaScript) in a clean MVC (Model View Controller) way.

7) It cross-browser Compatibility. Angular JS handles JavaScript code suitable for each browser.

Overall above Points I can Say:

** Angular JS is open Source framework to build large scale and high performance web application while keeping them as easy-to-maintain.

Core Features: -

1) Data-Binding: It automatic synchronization of data between model and view Components.
2) Scope: These are object that refers to Model. They act as glue between Controller and view.
3) Controller: It is JavaScript Function that are bounds to particular Scope.
4) Services: Angular JS come with several Built-in Services for example $http to make an XMLHttpRequests. These are singleton object which are Instantiated only once in App.
5) Filters: These select a subset of item from an array and return into new Array.
6) Directives: Directives are markers for DOM elements like (Elements, attributes, CSS and more). It is used for create Custom Html Tag that serves as new, custom Widgets. Angular JS has built-in directives like (ng-app, ng-model).
7) Template: These are the rendered view with information from the controller and model. It can be single file (index.html) or multiple view with one page using 'Partial'.
8) Routing: It is concept of switching Views.
9) Model View Whatever: MVC is used for design pattern for dividing application into different parts like (Model, view, Controller). Angular JS doesn't need MVC structure in traditional sense, but it rather something close to MV-VM (Model View or View Model). Model View Whatever.

10) Deep linking: Deep Linking allows to encode the application state in URL so that can be bookmarked. The application can be restored from the URL to the same state.
11) Dependency Injection: Angular JS has built-in dependency Injection subsystem that helps the developers by making application easy to develop, test and understand.

Advantages of AngularJS:-

1) AngularJS provides capability to create Single Page Application in a very clean and maintainable way.
2) AngularJS provides data binding capability to HTML thus giving user a rich and responsive experience
3) AngularJS code is unit testable.
4) AngularJS uses dependency injection and make use of separation of concerns.
5) AngularJS provides reusable components. With AngularJS, developer write less code and get more functionality.
6) In AngularJS, views are pure html pages, and controllers written in JavaScript do the business processing.
7) On top of everything, AngularJS applications can run on all major browsers and smart phones including Android and iOS based phones/tablets.

Disadvantages of AngularJS:-

Though AngularJS comes with lots of plus points but same time we should consider the following points−

Not Secure − Being JavaScript only framework, application written in AngularJS are not safe. Server side authentication and authorization is must to keep an application secure.

Not degradable − If your application user disables JavaScript then user will just see the basic page and nothing more.

Angular JS Components:

They are three main part of components:
1) **Ng-app**: This directive defines and links to an Angular JS application to HTML.
2) **Ng-model**: This directive binds the values of Angular JS Application data to HTML Input controls.
3) **Ng-binds**: This Directives binds the Angular JS data to HTML Tag.

1) **Point to Angular JS App:** When we place ng-app at that time Angular JS application start/ link to html page.[Example: <body ng-app = “myapp” :So here Module is myapp:]

2) **View:**
So below Syntax it say View.
a) Ng-controller tells to Angular JS what controller to use with this view.
b) helloTo.title tells to Angular JS write Model value to HTML page at this location.

```
View : <div ng-controller = "HelloController" >

  <h2>Welcome {{helloTo.title}} to the world of Tutorialspoint!</h2>

</div>
```

3) **Controller:** Below Syntax, Here the code registered with controller function name called `HelloController‘ in angular module named myapp. So Module and Controller can learn in next chapter with is marked in Light Blue Color. So here Controller function [HelloController] is registered in angular via angular module() and .controller() function call.

$Scope: is a Parameter passed to controller function is called `Model’. The controller function add helloTo JavaScript object, and in that Object it add a title field.

```
<script>
  angular.module("myapp", [])

  .controller("HelloController", function($scope) {
    $scope.helloTo = {};
    $scope.helloTo.title = "AngularJS";
  });
</script>
```

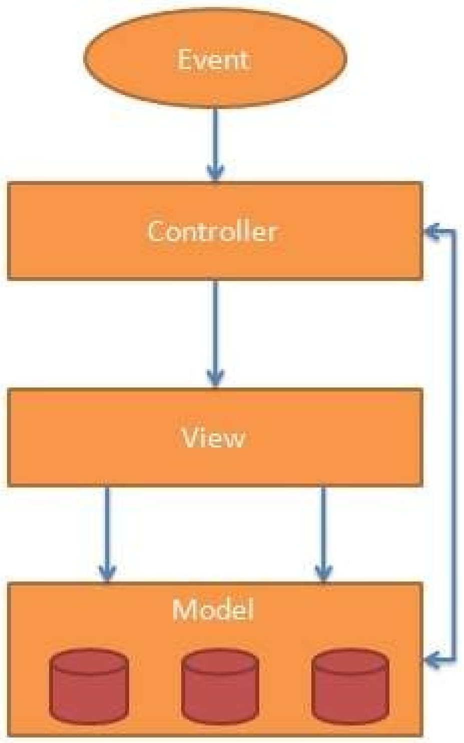
4) **When the page is loaded in the browser, following things happen −**

- HTML document is loaded into the browser, and evaluated by the browser. AngularJS JavaScript file is loaded, the angular *global* object is created. Next, JavaScript which registers controller functions is executed.

- Next AngularJS scans through the HTML to look for AngularJS apps and views. Once view is located, it connects that view to the corresponding controller function.

- Next, AngularJS executes the controller functions. It then renders the views with data from the model populated by the controller. The page is now ready.

Model View Controller or MVC as it is popularly called, is a software design pattern for developing web applications. A Model View Controller pattern is made up of the following three parts −



Model − It is the lowest level of the pattern responsible for maintaining data.

View − It is responsible for displaying all or a portion of the data to the user.

Controller − It is a software Code that controls the interactions between the Model and View.

MVC is popular because it isolates the application logic from the user interface layer and supports separation of concerns. The controller receives all requests for the application and then works with the model to prepare any data needed by the view. The view then uses the data prepared by the controller to generate a final presentable response. The MVC abstraction can be graphically represented as follows.

AngularJS MVC
The Model
The model is responsible for managing application data. It responds to the request from view and to the instructions from controller to update itself.

The View
A presentation of data in a particular format, triggered by the controller's decision to present the data. They are script-based template systems such as JSP, ASP, PHP and very easy to integrate with AJAX technology.

The Controller
The controller responds to user input and performs interactions on the data model objects. The controller receives input, validates it, and then performs business operations that modify the state of the data model.

AngularJS is a MVC based framework. In the coming chapters, we will see how AngularJS uses MVC methodology.

How AngularJS integrates with HTML:

1) Ng-app: This directives defines the starting the Angular JS application.
2) Ng-model: This directives creates model variable called “Anything” which can be used in html page and within <div> the ng-app located.
3) Ng-binds: This directives hold the model Variable value and display the text when the user type in textbox.
4) Closing </div> tag indicates Angular JS Application Closed.

AngularJS – Directives:
Angular JS Directives are used to extend HTML Element. These are special attributes start with ng-prefix. Some directives are
1) ng-app: These directive start the Angular JS Application
2) ng-init: These directive initialize application Data
3) ng-model: These directive defines model that is a variable in angular JS.
4) ng-repeat: These directive repeat HTML element for each item in angular JS.

Ng-app: Ng-App start the angular JS Application. It defines the root element. It automatically initialize and start the application when web page containing Angular JS Application Is loaded. It also used to load various Angular JS Modules in angular JS Application.

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

  ...

</div>
```

Ng-init: Ng-init Directive initialize the Angular JS Application Data. It put the values to variable to be used in the application.

```
<div ng-app = "" ng-init = "countries = [{locale:'en-US',name:'United States'}, {locale:'en-GB',name:'United Kingdom'}, {locale:'en-FR',name:'France'}]">

  ...

</div>
```

Ng-model: Ng-model directive defines the model/variable to be used in Angular JS Apps.

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

  ...

  <p>Enter your Name: <input type = "text" ng-model = "name"></p>

</div>
```

Ng-Repeat: Ng-repeat directive is to create HTML Element for each item when the application data is need an element.

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

  ...

  <p>List of Countries with locale :< /p>


  <ol>

    <li ng-repeat = "country in countries">

      {{ 'Country: ' + country.name + ', Locale: ' + country.locale }}

    </li>

  </ol>


</div>
```

Example for Angular JS – Directives:

```
<html>
```

```
   <head>
      <title>AngularJS Directives</title>
   </head>

   <body>
      <h1>Sample Application</h1>

      <div ng-app = "" ng-init = "countries = [{locale:'en-US',name:'United States'}, {locale:'en-
GB',name:'United Kingdom'}, {locale:'en-FR',name:'France'}]">
         <p>Enter your Name: <input type = "text" ng-model = "name"></p>
         <p>Hello <span ng-bind = "name"></span>!</p>
         <p>List of Countries with locale:</p>

         <ol>
            <li ng-repeat = "country in countries">
               {{ 'Country: ' + country.name + ', Locale: ' + country.locale }}
            </li>
         </ol>
      </div>

      <script src = "http://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js"></script>

   </body>
</html>
```

Angular JS – Expression:

Expressions are used to bind application data to html. Expressions are written inside double braces like {{expression}}. Expressions behaves in same way as ng-bind directives. AngularJS application expressions are pure JavaScript expressions and outputs the data where they are used.

Using Numbers: <p>Hello {{cost * product}} </p>

Using String: <p>Firstname: {{student.firstname}} </p>
<p>Lastname: {{student.lastname}} </p>

Using Object: <p>Roll No: {{student.rollno}} </p>

Using Array: <p>marks; {{marks [4]}} </p>.

```
<html>


  <head>

    <title>AngularJS Expressions</title>

  </head>


  <body>

    <h1>Sample Application</h1>


    <div ng-app = "" ng-init = "quantity = 1;cost = 30; student =
{firstname:'Mahesh',lastname:'Parashar',rollno:101};marks = [80,90,75,73,60]">

      <p>Hello {{student.firstname + " " + student.lastname}}!</p>

      <p>Expense on Books : {{cost * quantity}} Rs</p>
      <p>Roll No: {{student.rollno}}</p>

      <p>Marks(Math): {{marks[3]}}</p>

    </div>


    <script src = "http://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js"></script>

  </body>
</html>
```

Angular JS – Controller:
Angular JS Mainly relies on Controller to control the flow of data in application. The controller is defined using ng-controller directive. The controller is JavaScript object containing attribute/properties and function. Each controller accept $scope as a parameter which refers to application/module that controller is to control.

```
<div ng-app = "" ng-controller = "studentController">
  ...
</div>
```

Here we declared a controller ‘Student Controller’ using ng-controller directive.

```
<script>
  function studentController($scope) {
    $scope.student = {
      firstName: "Mahesh",
      lastName: "Parashar",

      fullName: function() {
        var studentObject;
        studentObject = $scope.student;
        return studentObject.firstName + " " + studentObject.lastName;
      }
    };
  }
</script>
```

1) StudentController is defined as a JavaScript Object with $scope as a argument.
2) $scope is reference to application which is to use the studentController object.
3) $scope.student is property for StudentController.
4) Firstname and lastname are property for $scope.student and Now $scope.student is Object.
5) Here fullname is a function of $scope.student object and whose task to return the combined name.
6) Fullname function we're getting the student object and then return the combined value.
7) As a note, we can also defined controller object in separate JS file.

Now we can use StudentController object in ng-model to get the values from controller by using the expression.

```
Enter first name: <input type = "text" ng-model = "student.firstName"><br>

Enter last name: <input type = "text" ng-model = "student.lastName"><br>

<br>

You are entering: {{student.fullName()}}
```

Here student.firstname and student.lastname were place to ng-model and in page it will show default values which is assigned in JS file.

And fullname() method can give in two ways a) expression or b) ng-binds.

Example for ng-controller (Angular JS Controller):

```
<html>


  <head>
    <title>Angular JS Controller</title>
    <script src = "http://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js"></script>
  </head>


  <body>
    <h2>AngularJS Sample Application</h2>


    <div ng-app = "mainApp" ng-controller = "studentController">
      Enter first name: <input type = "text" ng-model = "student.firstName"><br><br>
      Enter last name: <input type = "text" ng-model = "student.lastName"><br>
```

```
        <br>


        You are entering: {{student.fullName()}}
      </div>


      <script>
        var mainApp = angular.module("mainApp", []);


        mainApp.controller('studentController', function($scope) {
          $scope.student = {
            firstName: "Mahesh",
            lastName: "Parashar",


            fullName: function() {
              var studentObject;
              studentObject = $scope.student;
              return studentObject.firstName + " " + studentObject.lastName;
            }
          };
        });
      </script>

   </body>
</html>
```

Angular JS – Filters:
Filter are used to change modify the data and the clubbed into expression or directives using pipe character. Following is the list commonly used in directives.

1) Uppercase : It Convert the Text into Uppercase
2) Lowercase: it convert the text into lowercase
3) Currency: It place currency symbol before numeric value.
4) Filter: filter the array to a subset of it based on provided criteria.
5) Order BY: orders the array based on provided criteria.

Uppercase Filter:

Add uppercase filter to an expression using pipe character. Here we've added uppercase filter to print student name in all capital letters.

Lowercase Filter:
Add lowercase filter to an expression using pipe character. Here we've added lowercase filter to print student name in all lowercase letters.

Currency Filter:
Add currency filter to an expression returning number using pipe character. Here we've added currency filter to print fees using currency format.

Filter:
TO display only required user values.
Orderby:
To order values by numeric.

```
<html>


  <head>
    <title>Angular JS Filters</title>
    <script src = "http://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js"></script>
  </head>


  <body>
    <h2>AngularJS Sample Application</h2>
    <div ng-app = "mainApp" ng-controller = "studentController">
      <table border = "0">
        <tr>
          <td>Enter first name:</td>
          <td><input type = "text" ng-model = "student.firstName"></td>
        </tr>


        <tr>
          <td>Enter last name: </td>
          <td><input type = "text" ng-model = "student.lastName"></td>
        </tr>
        <tr>
          <td>Enter fees: </td>
          <td><input type = "text" ng-model = "student.fees"></td>
        </tr>

        <tr>
          <td>Enter subject: </td>
          <td><input type = "text" ng-model = "subjectName"></td>
        </tr>
      </table>
      <br/>

      <table border = "0">
        <tr>
          <td>Name in Upper Case: </td><td>{{student.fullName() | uppercase}}</td>
        </tr>

        <tr>
          <td>Name in Lower Case: </td><td>{{student.fullName() | lowercase}}</td>
        </tr>

        <tr>
          <td>fees: </td><td>{{student.fees | currency}}
          </td>
        </tr>

        <tr>
          <td>Subject:</td>

          <td>
            <ul>
              <li ng-repeat = "subject in student.subjects | filter: subjectName |orderBy:'marks'">
                {{ subject.name + ', marks:' + subject.marks }}
```

```
               </li>
            </ul>
         </td>
      </tr>
   </table>

</div>

<script>
   var mainApp = angular.module("mainApp", []);

   mainApp.controller('studentController', function($scope) {
      $scope.student = {
         firstName: "Mahesh",
         lastName: "Parashar",
         fees:500,

         subjects:[
            {name:'Physics',marks:70},
            {name:'Chemistry',marks:80},
            {name:'Math',marks:65}
         ],

         fullName: function() {
            var studentObject;
            studentObject = $scope.student;
            return studentObject.firstName + " " + studentObject.lastName;
         }
      };
   });
</script>

</body>
```

```
</html>
```

```
AngularJS Sample Application
Enter first name:
Enter last name:
Enter fees:
Enter subject:

Name in Upper Case:        MAHESH PARASHAR
Name in Lower Case:        mahesh parashar
fees:                      $500.00
                             • Math, marks:65
Subject:                     • Physics, marks:70
                             • Chemistry, marks:80
```

Angular JS – HTML DOM:

Following Directives bind the application data to attributes of Html Dom Elements.

1) ng-disabled
2) ng-show
3) ng-hide
4) ng-click

Ng-disabled: Add ng-disabled attribute to a HTML button and pass it a model. Bind the model to a checkbox and see the variation.

Ng-show: Add ng-show attribute to a HTML button and pass it a model. Bind the model to a checkbox and see the variation.

Ng-hide: Add ng-hide attribute to a HTML button and pass it a model. Bind the model to a checkbox and see the variation.

Ng-click: Add ng-click attribute to a HTML button and update a model. Bind the model to html and see the variation.

```
<html>

   <head>
      <title>AngularJS HTML DOM</title>
   </head>

   <body>
      <h2>AngularJS Sample Application</h2>
      <div ng-app = "">
```

```
         <table border = "0">
            <tr>
               <td><input type = "checkbox" ng-model = "enableDisableButton">Disable Button</td>
               <td><button ng-disabled = "enableDisableButton">Click Me!</button></td>
            </tr>

            <tr>
               <td><input type = "checkbox" ng-model = "showHide1">Show Button</td>
               <td><button ng-show = "showHide1">Click Me!</button></td>
            </tr>

            <tr>
               <td><input type = "checkbox" ng-model = "showHide2">Hide Button</td>
               <td><button ng-hide = "showHide2">Click Me!</button></td>
            </tr>

            <tr>
               <td><p>Total click: {{ clickCounter }}</p></td>
               <td><button ng-click = "clickCounter = clickCounter + 1">Click Me!</button></td>
            </tr>
         </table>

      </div>

      <script src = "http://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js"></script>

   </body>
</html>
```

AngularJS Sample Application

☐ Disable Button Click Me!

☐ Show Button

☐ Hide Button Click Me!

Total click: 12 Click Me!

Angular JS – Modules:
Angular JS support Modular approach. Modular is used to separate logic say service, application, controller etc. and it will keep the code clean. We define modular in separate JS file and the name them as per the module.js. They are two JS file one is Application and Controller JS file.

1) **Application Module:** Application module is used to initialize the application with controller.
2) **Controller Module:** Controller Module is used to define the controller.

Application Module:

[This array contain dependent Module]
Syntax: var mainapp = angular.module('myapp', []);

[Application Module Name ng-app='myapp'].

Here we've declared an application **myApp** module using [angular.module function]. We've passed an empty array to it. This array generally contains dependent modules.

Controller Module:

```
Syntax:  mainApp.controller("studentController", function($scope) {
  $scope.student = {
    firstName: "Mahesh",
    lastName: "Parashar",
    fees:500,
    subjects:[
      {name:'Physics',marks:70},
      {name:'Chemistry',marks:80},
      {name:'Math',marks:65},
      {name:'English',marks:75},
      {name:'Hindi',marks:67}
    ],

    fullName: function() {
      var studentObject;
      studentObject = $scope.student;
      return studentObject.firstName + " " + studentObject.lastName;
    }
  };
});
```

Here we declared Controller StudentController module to mainapp.controller() Function.

Use Modules :

```
<div ng-app = "mainApp" ng-controller = "studentController">
  ...
  <script src = "mainApp.js"></script>
  <script src = "studentController.js"></script>

</div>
```

Here we've used application module using ng-app directive and controller using ng-controller directive. We've imported mainApp.js and studentController.js in the main html page.

Angular JS – Forms:

Angular JS enriches form filling and validation. Here we use ng-click event for angular JS click button and use **$dirty** and **$invalid** for validation in seamless way. In form validation we use **“novalidation”** to disable browser specific validation. Form controls make heavy use on Angular JS Events.

Events:

Ng-click

Ng-dbl-click

Ng-mouseenter

Ng-mouseleave

Ng-moverhover

Ng-mouseup

Ng-mousedown

Ng-mousemove

Ng-keyup

Ng-keydown

Ng-change

Ng-keypress

Validate Data:

Following can be used to track error:

1) **$dirty:** states the value has changed.

2) **$invalid:** states that value entered is invalid

3) **$error:** states that exact error.

```
<html>
  <head>
    <title>Angular JS Forms</title>
    <script src = "http://ajax.googleapis.com/ajax/libs/angularjs/1.3.14/angular.min.js"></script>

    <style>
      table, th , td {
        border: 1px solid grey;
        border-collapse: collapse;
        padding: 5px;
      }

      table tr:nth-child(odd) {
        background-color: #f2f2f2;
      }

      table tr:nth-child(even) {
        background-color: #ffffff;
      }
    </style>

  </head>
  <body>

    <h2>AngularJS Sample Application</h2>
    <div ng-app = "mainApp" ng-controller = "studentController">
```

```
      <form name = "studentForm" novalidate>

         <table border = "0">

            <tr>

               <td>Enter first name:</td>

               <td><input name = "firstname" type = "text" ng-model = "firstName" 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>

               </td>

            </tr>


            <tr>

               <td>Enter last name: </td>

               <td><input name = "lastname"  type = "text" ng-model = "lastName" 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>

               </td>

            </tr>


            <tr>

               <td>Email: </td><td><input name = "email" type = "email" ng-model = "email" length =
"100" required>

                  <span style = "color:red" ng-show = "studentForm.email.$dirty && studentForm.email.
$invalid">

                     <span ng-show = "studentForm.email.$error.required">Email is required.</span>

                     <span ng-show = "studentForm.email.$error.email">Invalid email address.</span>

                  </span>

               </td>

            </tr>
            <tr>
               <td>
                  <button ng-click = "reset()">Reset</button>
               </td>
               <td>
                  <button ng-disabled = "studentForm.firstname.$dirty &&
                     studentForm.firstname.$invalid || studentForm.lastname.$dirty &&
                     studentForm.lastname.$invalid || studentForm.email.$dirty &&
                     studentForm.email.$invalid" ng-click="submit()">Submit</button>
               </td>
            </tr>

         </table>
      </form>
   </div>

   <script>
      var mainApp = angular.module("mainApp", []);

      mainApp.controller('studentController', function($scope) {
         $scope.reset = function(){
            $scope.firstName = "Mahesh";
            $scope.lastName = "Parashar";
            $scope.email = "MaheshParashar@tutorialspoint.com";
         }

         $scope.reset();
      });
   </script>

   </body>
</html>
```