Week 6:

Write an XML file which will display the Book information which includes the following: Title of the book, Author Name, ISBN number, Publisher name, Edition, Price.

i. Write a Document Type Definition (DTD) to validate the above XML file.
ii. Write a XML Schema Definition (XSD) to validate the above XML file.
iii. Write a Extensible Stylesheet Language Transformations (XSL) to display data in table format

**Description:**

**DTD:**

- The document type definition used to define the basic building block of any xml
- document.
- Using DTD we can specify the various elements types, attributes and their relationship with one another.
- Basically DTD is used to specify the set of rules for structuring data in any XML file.
- ***Various building blocks of XML are***
- 1. Elements
- 2. Attribute
- 3. CDATA
- 4. PCDATA
- CDATA also means character data.
- CDATA is text that will not be parsed by a parser.
- Tags inside the text will not be treated as markup and entities will not be expanded.
- PCDATA means parsed character data.
- Character data is e the text found between the start tag and the of an XML document.

- PCDATA is text that will be parse by a parser Tags inside the text will be treated as markup and entities will be expanded.
- <!ELEMENT student (name)*>
- * indicates zero or more occurrences.
- + indicates 1,2,…n
- ? indicate 0,1
- Default exactly once

**XSD:**

- The <schema> element is the root element of every XML schema.
- <schema> elementmay contains some attributes.
- <?xml version = "1.0"?>
- <xs:schema>
- -------------
- </xs:schema>
- XSD elements are two types.
- 1\. Simple element
- 2\. Complex element
- Various data types are String, Date, Numeric, Boolean
- xs:string It containing group of characters, lines, tabs or white spaces.
- xs:date Used to represent date. The format of this date is yyyy-mm-dd
- xs:time use to represent time. The format of this time is hh:mm:ss
- xs:decimal use to represent float values.
- xs:integer use to represent integer values.
- xs:boolean used to resent Boolean values either true or false.

## XSLT:

- XSLT is a language capable of transforming as well as formatting given XML document into required format such as html and pdf files
- XSL means eXtensible Stylesheet Language
- Start with a raw XML document (create a xml file)
- Create an XSL style sheet (create a xsl file)
- Link the XSL style sheet to the XML document
- Use XSLT processors to transform your XML into XHTML

## Programs:

### book.dtd:

```
<!ELEMENT bookdetails (books+)>
<!ELEMENT books (title,author,isbn,publisher,edition,price)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT isbn (#PCDATA)
<!ELEMENT publisher (#PCDATA)
<!ELEMENT edition (#PCDATA)>
<!ELEMENT price (#PCDATA)>
```

### book.xml:

```
<?xml version="1.0"?>
<!DOCTYPE bookdetails SYSTEM "book.dtd">
<?xml:stylesheet type="text/xsl" href="book.xsl"?>
<bookdetails>
<books>
<title> C </title>
<author> Balaguruswamy </author>
<isbn> 123456</isbn>
```

```
<publisher>Oxford</publisher>
<edition> Edition-II </edition>
<price>$30.00</price>
</books>
<books>
<title> C++ </title>
<author> yaswanth kanethkar </author>
<isbn>654321</isbn>
<publisher>Tata McHill</publisher>
<edition> Edition-I </edition>
<price>$35.00</price>
</books>
<books>
<title> JAVA </title>
<author> Herbert Schildt </author>
<isbn>789456</isbn>
<publisher>Cenage</publisher>
<edition> Edition-IV </edition>
<price>$50.00</price>
</books> </bookdetails>
```

**book.xsd:**

```
<?xml version="1.0"?>

<xs:schema>

xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="bookdetails">

<xs:complexType>

<xs:sequence>

<xs:element name="books" type="xs:string"/>

<xs:element name="title" type="xs:string"/>

<xs:element name="author" type="xs:string"/>

<xs:element name="isbn" type="xs:string"/>

<xs:element name="publisher" type="xs:string"/>
```

```
<xs:element name="edition" type="xs:string"/>

<xs:element name="price" type="xs:string"/>

</xs:sequence>

</xs:complexType>

</xs:element>

</xs:schema>
```

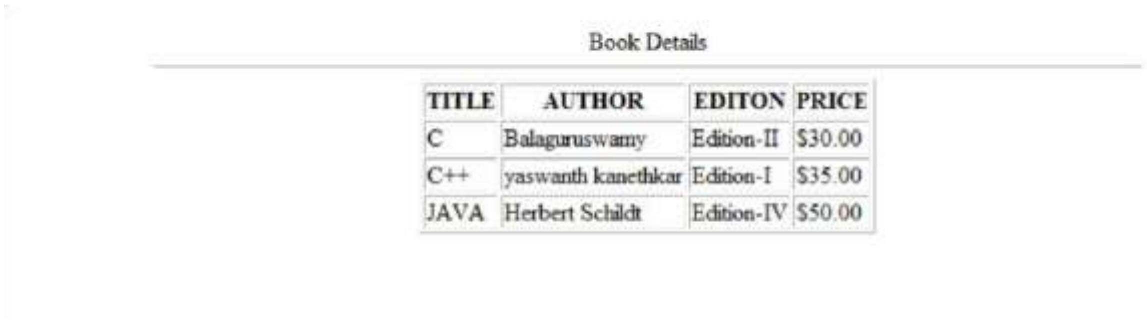
**book.xml:**

```
<?xml version="1.0"?>
<!DOCTYPE bookdetails SYSTEM "book.dtd">
<?xml:stylesheet type="text/xsl" href="book.xsl"?>
<bookdetails>
<books>
<title> C </title>
<author> Balaguruswamy </author>
<isbn> 123456</isbn>
<publisher>Oxford</publisher>
<edition> Edition-II </edition>
<price>$30.00</price>
</books>
<books>
<title> C++ </title>
<author> yaswanth kanethkar </author>
<isbn>654321</isbn>
<publisher>Tata McHill</publisher>
<edition> Edition-I </edition>
<price>$35.00</price>
</books>
<books>
<title> JAVA </title>
<author> Herbert Schildt </author>
<isbn>789456</isbn>
<publisher>Cenage</publisher>
<edition> Edition-IV </edition>
<price>$50.00</price>
</books> </bookdetails>
```

**book.xsl:**

```
<?xml version="1.0"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
<head> <center>Book Details</center> </head>
<body>
<hr width="50%"/>
<table border="1" align="center">
<tr>
<th> TITLE </th>
<th> AUTHOR </th>
<th>ISBN</th>
<th>PUBLISHER</th>
<th> EDITON </th>
<th> PRICE </th>
</tr>
<xsl:for-each select="bookdetails/books">
<tr>
<td> <xsl:value-of select="title"/></td>
<td> <xsl:value-of select="author"/></td>
<td> <xsl:value-of select="isbn"/></td>
<td> <xsl:value-of select="publisher"/></td>
<td> <xsl:value-of select="edition"/></td>
<td> <xsl:value-of select="price"/></td>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

## Output:



Book Details

| TITLE | AUTHOR | EDITON | PRICE |
|---|---|---|---|
| C | Balaguruswamy | Edition-II | $30.00 |
| C++ | yaswanth kanethkar | Edition-I | $35.00 |
| JAVA | Herbert Schildt | Edition-IV | $50.00 |

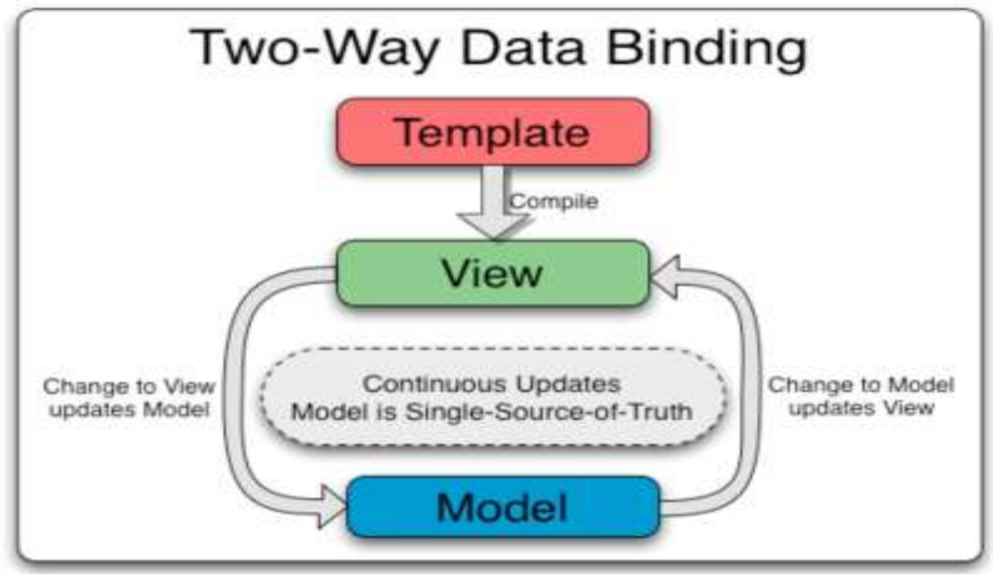
Week 7:

Implement the following in Angular JS

i. data binding.
ii. directives and Events.
iii. includes.

**Data Binding:**

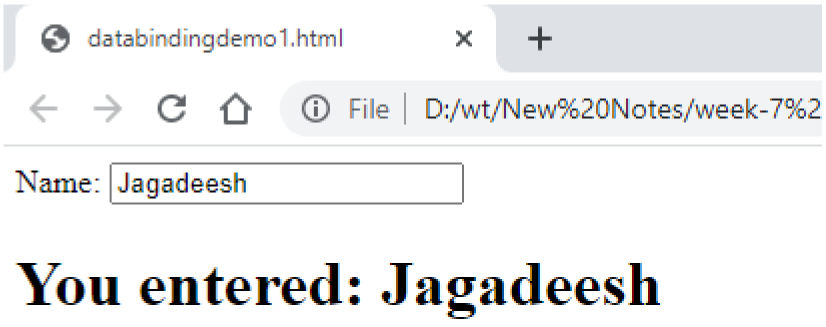
- Data binding is a very useful and powerful feature used in software development technologies.
- It acts as a bridge between the view and business logic of the application.
- AngularJS follows Two-Way data binding model.
- Data-binding in Angular apps is the automatic synchronization of data between the model and view components.
- Data binding lets you treat the model as the single-source-of-truth in your application.
- The view is a projection of the model at all times.
- If the model is changed, the view reflects the change and vice versa.


Two-Way Data Binding
Template
Compile
View
Change to View updates Model
Continuous Updates Model is Single-Source-of-Truth
Change to Model updates View
Model

**Program:**

```
<!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">

Name: <input ng-model="name">

<h1>You entered: {{name}}</h1>

</div>

<script>

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

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

   $scope.name = "Jagadeesh";
});
</script>
</body>
</html>
```

**Output:**



Name: Jagadeesh

You entered: Jagadeesh

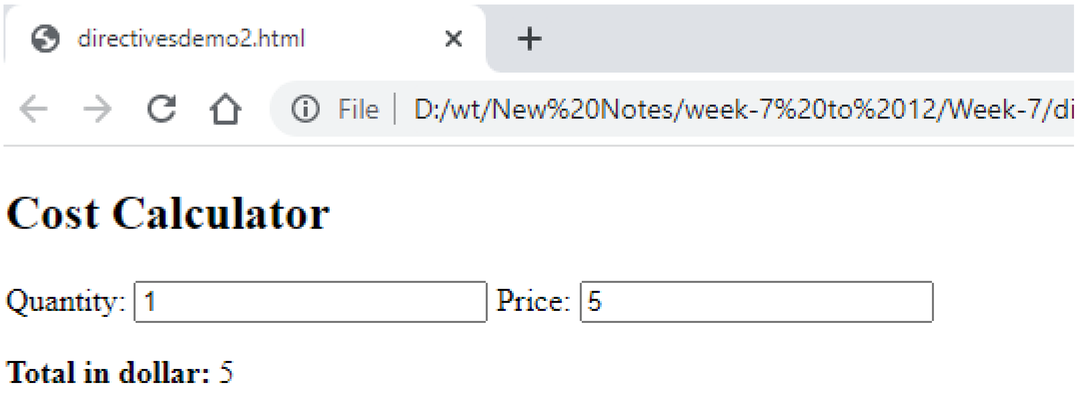
**Directives:**

- AngularJS directives are extended HTML attributes with the prefix ng-.
- The ng-app directive initializes an AngularJS application.
- The ng-init directive initializes application data.
- The ng-model directive binds the value of HTML controls (input, select, textarea) to application data.

**Program:**

```
<!DOCTYPE html>

<html>

<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>

<body>

<div data-ng-app="" data-ng-init="quantity=1;price=5">

<h2>Cost Calculator</h2>

Quantity: <input type="number" ng-model="quantity">

Price: <input type="number" ng-model="price">

<p><b>Total in dollar:</b> {{quantity * price}}</p>


</div>

</body>

</html>
```

**Output:**



Cost Calculator

Quantity: 1 Price: 5

**Total in dollar:** 5

**Events:**

You can add AngularJS event listeners to your HTML elements by using one or more of these directives:

- ng-blur
- ng-change
- ng-click
- ng-copy
- ng-cut
- ng-dblclick
- ng-focus
- ng-keydown
- ng-keypress
- ng-keyup
- ng-mousedown
- ng-mouseenter
- ng-mouseleave
- ng-mousemove
- ng-mouseover
- ng-mouseup
- ng-paste

The event directives allows us to run AngularJS functions at certain user events.

An AngularJS event will not overwrite an HTML event, both events will be executed.

$event Object

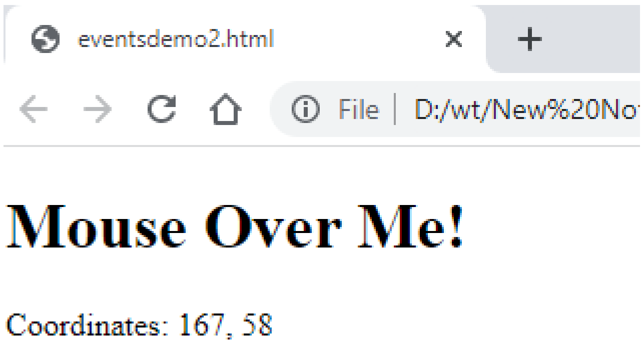
You can pass the $event object as an argument when calling the function.

The $event object contains the browser's event object

**Program:**

```
<!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">

<h1 ng-mousemove="myFunc($event)">Mouse Over Me!</h1>

<p>Coordinates: {{x + ', ' + y}}</p>

</div>

<script>

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

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

   $scope.myFunc = function(myE) {

       $scope.x = myE.clientX;

       $scope.y = myE.clientY;

   }

});

</script>

</body>

</html>
```

**Output:**



Mouse Over Me!

Coordinates: 167, 58

**Include:**

With AngularJS, you can include HTML content using the **ng-include** directive

The HTML files you include with the ng-include directive, can also contain AngularJS code

**Program:**

**Main.html:**

```
<html>

<head></head>

<body>

<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>Name: </td>
    <td>{{student.fullName()}}</td>
  </tr>
</table>
</body>
</html>
```

**Subjects.html:**

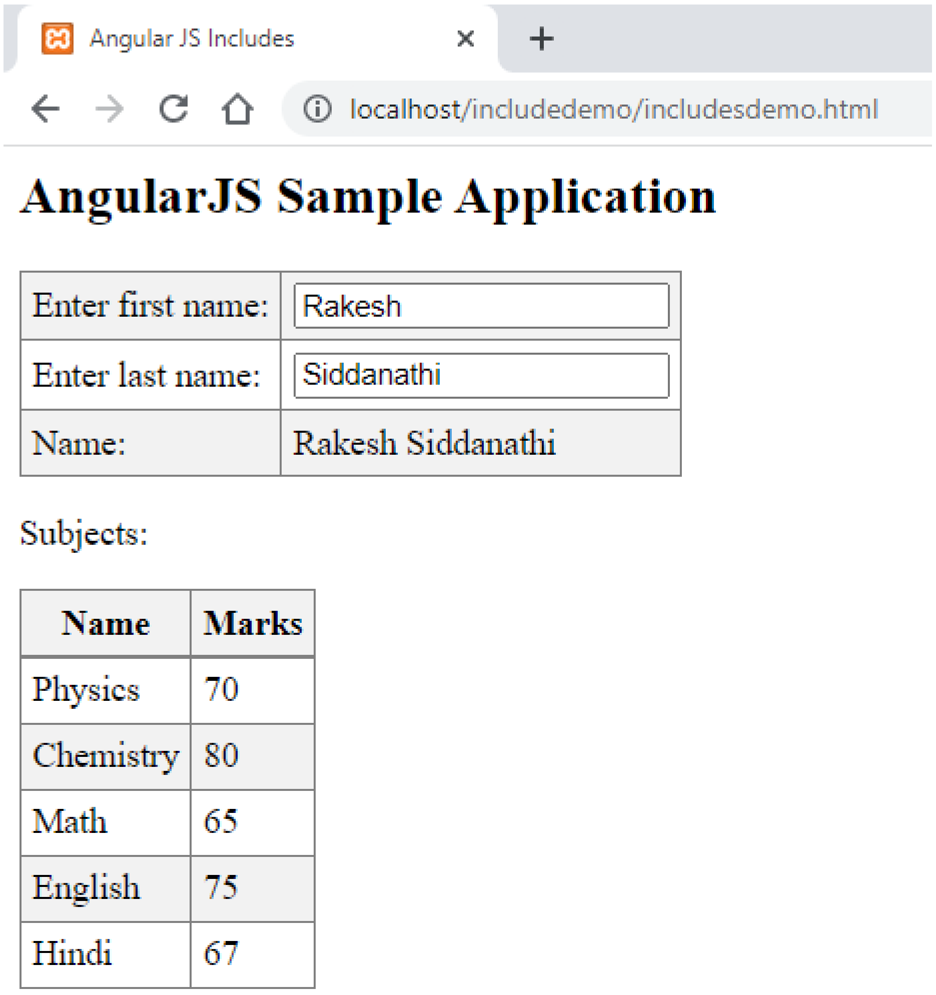
```
<html>
<head></head>
<body>
<p>Subjects:</p>
<table>
  <tr>
    <th>Name</th>
    <th>Marks</th>
  </tr>

  <tr ng-repeat = "subject in student.subjects">
```

```
    <td>{{ subject.name }}</td>

    <td>{{ subject.marks }}</td>

  </tr>

</table>

</body>

</html>
```

**<u>Includesdemo.html:</u>**

```
<html>

  <head>

    <title>Angular JS Includes</title>

    <script src = "https://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">
         <div ng-include = "'/includedemo/main.html'"></div>
         <div ng-include = "'/includedemo/subjects.html'"></div>
      </div>

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

         mainApp.controller('studentController', function($scope) {
            $scope.student = {
               firstName: "Rakesh",
               lastName: "Siddanathi",
               fees:50000,

               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;
              }
            };
          });
        </script>

    </body>
</html>
```

**<u>Output:</u>**



## AngularJS Sample Application

| | |
|---|---|
| Enter first name: | Rakesh |
| Enter last name: | Siddanathi |
| Name: | Rakesh Siddanathi |

Subjects:

| Name | Marks |
|---|---|
| Physics | 70 |
| Chemistry | 80 |
| Math | 65 |
| English | 75 |
| Hindi | 67 |

Week-8:

Implement the following in Angular JS

i. routing.

ii. form validation.

iii. fetching data from MySQL.

**Description:**

**Routing:**

If you want to navigate to different pages in your application, but you also want the application to be a SPA (Single Page Application), with no page reloading, you can use the ngRoute module.

The ngRoute module *routes* your application to different pages without reloading the entire application.

Use the $routeProvider to configure different routes in your application

We can also use the otherwise method, which is the default route when none of the others get a match.

**Program:**

**About.html:**

```
<html>

<head>

</head>

<body>

<p>Click on the below link o redirect to College info</p><br>

<a href="https://aec.edu.in/?p=About-AEC">About AEC</a>

</body>

</html>
```

**Aec.html:**

```
<html>

<head>

</head>

<body>

<p>Click on the below link to redirect to home page</p><br>

<a href="https://www.aec.edu.in">AEC</a>

</body>

</html>
```

**Library.html:**

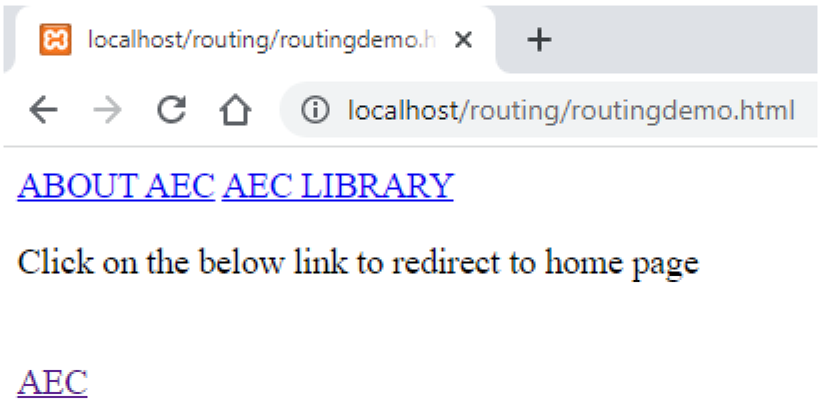
```
<html>

<head>

</head>

<body>

<p>Click on the below link to redirect to library info</p><br>

<a href="https://aec.edu.in/?p=Library">AEC-Library</a>

</body>

</html>
```

**Routingdemo.html:**

```
<!DOCTYPE html>

<html>

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

```
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular-
route.js"></script>

<body ng-app="myApp">

<a href="#!about-aec">ABOUT AEC</a>

<a href="#!aec-lib">AEC LIBRARY</a>

<div ng-view></div>

<script>

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

app.config(function($routeProvider) {

    $routeProvider

    .when("/about-aec", {

        templateUrl : "about.html"

    })

    .when("/aec-lib", {

        templateUrl : "library.html"

    })

    .otherwise( {

        templateUrl : "aec.html"

    });

});

</script>

</body>

</html>
```

**Output:**


localhost/routing/routingdemo.html

ABOUT AEC AEC LIBRARY

Click on the below link to redirect to home page

AEC

**Form Validation:**

AngularJS offers client-side form validation.

AngularJS monitors the state of the form and input fields (input, textarea, select), and lets you notify the user about the current state.

AngularJS also holds information about whether they have been touched, or modified, or not.

You can use standard HTML5 attributes to validate input, or you can make your own validation functions.

AngularJS is constantly updating the state of both the form and the input fields.

Input fields have the following states:

- $untouched The field has not been touched yet
- $touched The field has been touched
- $pristine The field has not been modified yet
- $dirty The field has been modified
- $invalid The field content is not valid
- $valid The field content is valid

They are all properties of the input field, and are either true or false.

Forms have the following states:

- $pristine No fields have been modified yet
- $dirty One or more have been modified
- $invalid The form content is not valid
- $valid The form content is valid
- $submitted The form is submitted

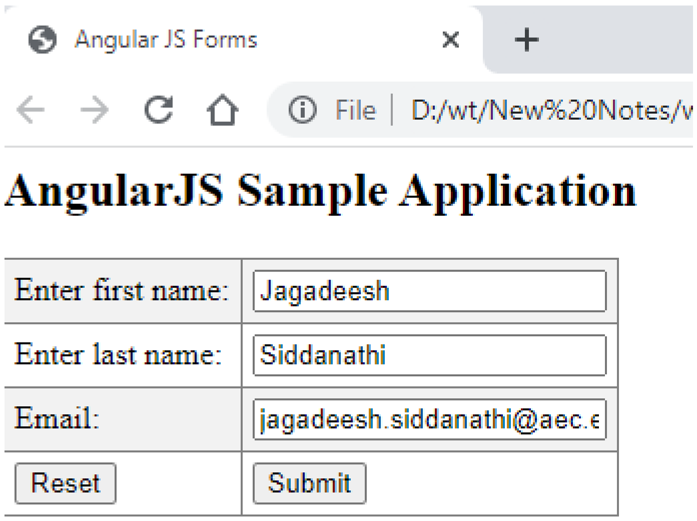
They are all properties of the form, and are either true or false.

**Program:**

```
<html>

  <head>

    <title>Angular JS Forms</title>

    <script src =
"https://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>
```

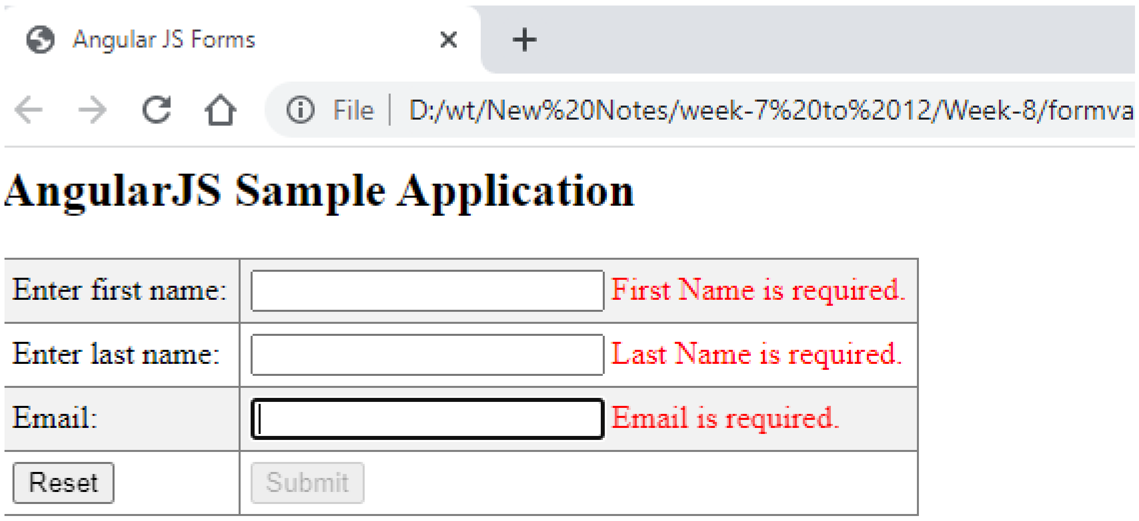
```
    </div>
    <script>
      var mainApp = angular.module("mainApp", []);
      mainApp.controller('studentController', function($scope) {
        $scope.reset = function() {
          $scope.firstName = "Jagadeesh";
          $scope.lastName = "Siddanathi";
          $scope.email = "jagadeesh.siddanathi@aec.edu.in";
        }
        $scope.reset();
      });
    </script>
  </body>
</html>
```

**Output:**



# AngularJS Sample Application

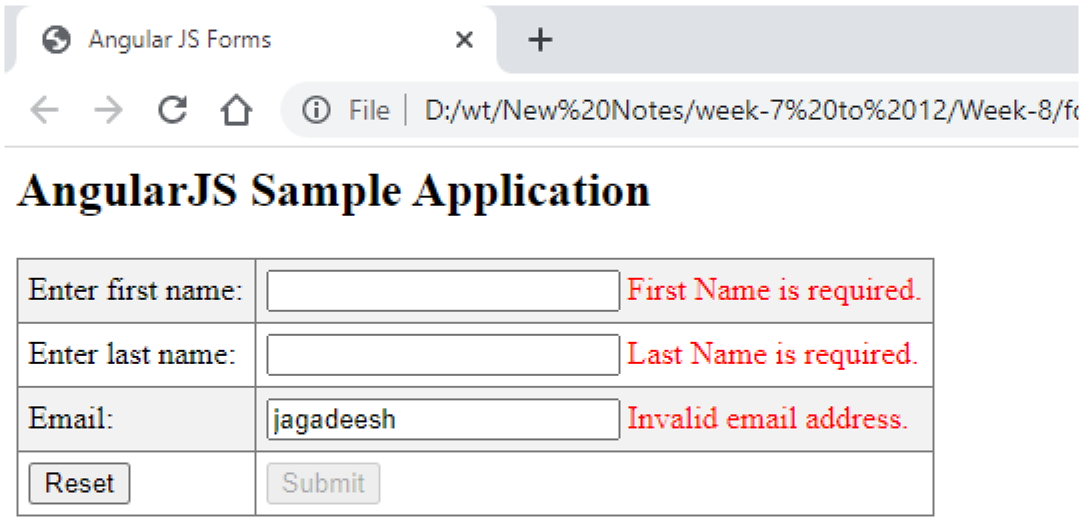
| | |
|---|---|
| Enter first name: | Jagadeesh |
| Enter last name: | Siddanathi |
| Email: | jagadeesh.siddanathi@aec.e |
| Reset | Submit |


Angular JS Forms

File | D:/wt/New%20Notes/week-7%20to%2012/Week-8/formva

## AngularJS Sample Application

| | | |
|---|---|---|
| Enter first name: | | First Name is required. |
| Enter last name: | | Last Name is required. |
| Email: | | Email is required. |
| Reset | Submit | |


Angular JS Forms

File | D:/wt/New%20Notes/week-7%20to%2012/Week-8/fo

## AngularJS Sample Application

| | | |
|---|---|---|
| Enter first name: | | First Name is required. |
| Enter last name: | | Last Name is required. |
| Email: | jagadeesh | Invalid email address. |
| Reset | Submit | |

**Fetching Data from MySQL:**

**Program:**

**Data.html:**

```
<!DOCTYPE html>
```

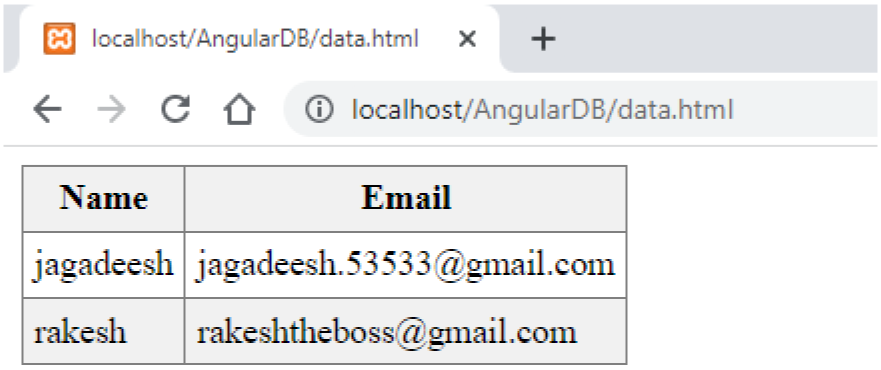
```
<html >

<style>

table, th , td  {

  border: 1px solid grey;

  border-collapse: collapse;

  padding: 5px;

}

table tr:nth-child(odd) {

  background-color: #f1f1f1;

}

table tr:nth-child(even) {

  background-color: #ffffff;

}

</style>

<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>

<body>


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



<table>

  <th>Name</th>

  <th>Email</th>

  <tr ng-repeat="x in users">

    <td>{{ x.Name }}</td>
    <td>{{ x.Email }}</td>
  </tr>
</table>

</div>

<script>
var app = angular.module('myApp', []);
app.controller('customersCtrl', ['$scope', '$http', function ($scope, $http) {
 $http({
  method: 'get',
  url: 'connect.php'
 }).then(function successCallback(response) {
  // Store response data
  $scope.users = response.data.records;
 });
}]);
</script>
</body>
</html>
```

**<u>Connect.php:</u>**

```
<?php
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");
```

```
$conn = new mysqli("localhost", "root", "", "aditya");


$result = $conn->query("SELECT uname,email FROM registration");


$outp = "";
while($rs = $result->fetch_array(MYSQLI_ASSOC)) {
  if ($outp != "") {$outp .= ",";}
  $outp .= '{"Name":"'  . $rs["uname"] . '",';
  $outp .= '"Email":"'   . $rs["email"]             . '"}';
}
$outp ='{"records":['.$outp.']}';
$conn->close();


echo($outp);
?>
```

**Output:**



| Name | Email |
|---|---|
| jagadeesh | jagadeesh.53533@gmail.com |
| rakesh | rakeshtheboss@gmail.com |