Angular Testing

- Angular supports frameworks like MVC and MVVM.
- MVC and MVVM enable Unit Testing.
- They support **Test Driven Development** [TDD].
- Unit Testing include testing every function that you write for component, pipe, service etc.
- Testing verifies whether the expected values and return value are same and will report bugs.
- Angular is integrated with Jasmine & Karma, which are used for unit testing. [Protractor, xUnit, etc]

Setup Framework for Testing [Manually]

- Create a new folder in your workspace by name "TestingDemo"
- Install Jasmine Framework for your workspace
  > npm install jasmine-core
- Create a new HTML document and add into project and link the following files
  Home.html

```
<!DOCTYPE html>
<html>
   <head>
      <link rel="stylesheet"
href="../node_modules/jasmine-core/lib/jasmine-
core/jasmine.css">
      <script src="../node_modules/jasmine-
core/lib/jasmine-core/jasmine.js"></script>
      <script src="../node_modules/jasmine-
core/lib/jasmine-core/jasmine-html.js"></script>
```

```
        <script src="../node_modules/jasmine-core/lib/jasmine-core/boot.js"></script>
    </head>
</html>
```

- **Testing every component, service, pipe or module function include 3 stages**
    - Arrange
    - Act
    - Assert
- Arrange: It is describing the test. **[describe()]**
- Act: It specifies the function to test. **[it()]**
- Assert: Verifying the expected and return value. Report the test result pass or fail. **[expect()]**

Syntax:

```
describe("Test Description", function(){
    it("Test Function and It Purpose", function(){
        expect(returnedValue).toBe(ExpectedValue);
    })
})
```

Ex:

- Add a new folder "Components"
- Add following files

  Home.component.js

```
function addition(a, b)
{
    return a + b;
```

```
}
function hello(str)
{
   return str;
}
```

Home.component.spec.js

```
describe("Math Component Test", function(){
   it("Addition Test", function(){
      expect(addition(20,20)).toBe(40);
   })
   it("Hello Test", function(){
      expect(hello("john")).toBe("johns");
   })
})
```

Home.html

```
<!DOCTYPE html>

<html>

   <head>

      <link rel="stylesheet"
href="../node_modules/jasmine-core/lib/jasmine-
core/jasmine.css">

      <script src="../node_modules/jasmine-
core/lib/jasmine-core/jasmine.js"></script>

      <script src="../node_modules/jasmine-
core/lib/jasmine-core/jasmine-html.js"></script>
```

```
    <script src="../node_modules/jasmine-core/lib/jasmine-core/boot.js"></script>
    <script src="Components/math.component.js"></script>
    <script src="Components/math.component.spec.js"></script>
  </head>
</html>
```

- Go to “karma.config” to configure the live server and browser for testing.
- In your angular workspace run the command
  > ng test --project=yourProjectName

Building and Deploying

- Building is the process of compiling Angular application.
- Checking the syntax errors, Dependencies, Directives, Meta data, Services etc.
- After checking trans compiling Typescript into JavaScript.
- Copying all compile JavaScript file into “output directory”.
- Final build information is copied into “dist” folder.
- “dist” is used for deploying.
- Earlier you were using “**ng serve**” which is implicitly using a “web pack” for building while serving.
- Angular project command for building application

Command	Description
ng build	It builds complete angular application with defaults. - Default output directory is “dist” - Default startup page is “index.html” - Default base href is “/”
ng build --outputPath=folderName	It build the output int a new directory instead of “dist”
ng build --baseHref=http://locahost:8080	- To change the base href.
ng build --index=login	- To change the startup page

Deploying Angular Application

- You can deploy angular application on Local Servers
  - XAMPP
  - IIS
  - WAMP
  - MAMP etc.
- You can deploy angular application on cloud servers
  - Firebase
  - Azure
  - AWS

    - Now
    - Netify
    - GitHub Page
    - NPM etc.
- Angular can use manual and automated deployment tools
- Always recommended to use automated deployment tools
    - @angular/fire- firebase
    - @azure/ng-deploy
    - @zeit/ng-deploy
    - Angular-cli-ghpages [Git]
    - Ngx-deploy-npm

Deploying on Firebase:

- Create a firebase account by using your Google account Firebase.google.com
- Create a new Project: smart-shop
- Go to your project workspace in VS code
- **Install firebase tools for your PC**
  > npm install -g firebase-tools
- **Login into firebase account**
  > firebase login
- **Build your project for production**
  > ng build --project=flipkart –prod
- This will generate “dist” folder with your project repository
- **Add angular firebase plugin for your project, which will take care about the deployment process.**

> ng add @angular/fire --project=flipkart
Select your firebase project to deploy: smart-shop

- **Deploy entire repository into live application**
  > ng deploy --project=flipkart

Note: Deployment finished then go to “hosting” and access the domain