Session 1&2

**Node**

Node JS is not a library, It’s a runtime env for JS.

Chrome V8 Engine

http ---- node ---- **npm**

|

URL etc

1. Download Node
2. Install
3. Open cmd

node –version

Node JS we get another command called npm (node package manager)

NPM-> Software App to get meta data of library sp that we can use in our application.

Maven Pack Manager🡺 java

Npm –version

To create angular project we must have internet connection for downloading ng modules

Npm install jquery

Npm install angular 🡪 1.7 (Angular JS)

**Angular**

**Angular** is open source web framework which helps in developing SPA (Single page aplication), part of google

Interpreter(JS) vs Compiler vs Transpiler

Ng (next generation)

Npm install @angular/cli -g

Ng –version

Command to create new project

Ng new project-name

Routing -> No

CSS -> YES

Cd project-name

To run the project:

Ng serve

Ng serve -o (to run on default browser)

By default, the port is 4200, but we can change it

Ng serve –port 1234 -o after compiling 100% it will automatically open on 1234 port in default browser.

<http://localhost:4200>

Open the project in VS Code

npm install : to install modules

ng build: Build the project in JS for PROD.

Developer(Angular+MongoDB) UNIT\_TEST(Funcnality/Feature) SIT(System Integration Testing) UAT(User Acceptance Testing) PROD(Angular + MongoDB (3)+Reporting)

7(Major Version).12(Minor Version).5(Patch)

**Directory Structure**

Src

App folder

App.component.ts: For TypeScript Code (logic for code)

App.componenet.html: here we can write static html

App.component.css: Styling of component goes here

Package.json -> It's npm's ecosystem and its responsible for keeping meta data about pacakges/libraries you use as your dependency.

Angular.json -> <https://medium.com/nerd-for-tech/what-is-inside-of-angular-json-file-636e81e67651>

Package-lock.json -> holds information on the dependencies or packages installed for a node. js project, including their exact version numbers.

**Dist (ng build)**

1. Runtime.js -> this is for webpack runtime.
2. Polyfill.js -> Helps to load new ESM version with old browsers.
3. Vendor.js -> This is our actual project code.
4. Main.js -> The points to first module of project.
5. Styles.js -> All CSS code goes here.

Session 3

**Angular Data Binding**

**Data Binding is used to share data b/w component to template(view).**

**Component 🡨-------------🡪 View (HTML)**

1. One way Binding
2. String Interpolation: Component to View

{{}}

{{variableName}}

{{5+4}}

{{display()}}

1. Property Binding: Component to View

[]

Name:<input type=”text” [value]=”variableName”/>

<p [innerText]=”variableName”></p>

Angular maintains the properties and attributes in sync when you use them.

In String Interpolation all data always consider as string only.

But in property binding apart from string we can use other data types.

Session 4

1. Event Binding: View to Component

Angular is using same event as provided by JS, the only difference in “on” pre-fixed and name must wrap with parenthesis.

JS Event Angular Event

onClick (click)

onDblClick (bdclick)

onMouseOver (mouseOver)

onSubmit (ngSubmit)

In JS we call normal normal JS function through Event.

But in Angular we are calling component class function through events.

1. Two Way Binding
2. Event Binding and String Interpolation or Property Binding is known as two way binding.
3. Two way binding uses ngModel attribute.

If we do any changes in view automatically it will update component and vice-versa.

[(ngModel)]

<input type=”text” [(ngModel)]=”name”/>

ngModel is predefined attribute part of FormsModule.

Session 5/6

**Types of Directives**

Created new project for learning directives

Directive: it is used to add extra behavior or functionality to html or DOM.

1. Component Directive: using this type of directive we can create the user defined tags.

@Component ({

Selector:”my-root”,

templateUrl:”app.html”,

stylesUrl:[“app.css”]

})

Export class AppComponent{

}

1. Structural Directive: using these directives we add or remove dom elements from HTML pages (view).

\*ngFor

\*ngIf

1. Attribute Directive: mainly used to apply CSS rules.

ngStyle -> like a style (inline CSS File)

ngClass -> like a CSS Class Selector

**Session 7&8**

**Angular Form**

Using angular Form we can pass the group of values in the form of json.

2 types of forms

1. Template driven form
2. Model driven form or Reactive form.

Template Driven form

1. The flow of the application from view (template) to component.
2. This type of forms easy to develop.
3. It is good for small application.

Reactive forms

1. The flow the application component to view or template.
2. To develop this this type of form people must have good knowledge on typescript.
3. It is good for enterprise application.

Created new project -> ng new angular-forms

in template driven form we have to create the reference of form.

Syntax

<form #loginRef=”ngForm”> </form>

Reference name start with # followed by reference name equal to ngForm.

ngForm is a pre-defined attribute, which help to create the reference. This attribute is a part of FormsModule. So we have to import this module in app.module.ts file mandatory.

If you want to bind textfield, password field to ngForm reference. We must use ngModel attribute.

Reactive form or Model Driven Form

According to Reactive form in html textfield, password field, radio button, checkbox is known as FormControl.

We can’t create FormControl without FormGroup.

So FormGroup is a collection of more than one FormControl.

In Angular FormControl and FormGroup are API.

In Login Page we require one FormGroup which contains two FormControls.

formGroup and formControlName are pre-defined attribute part of ReactiveFormsModule. So we have to import ReactiveFormsModule in app.module.ts file.

**Angular Service**

In case of any complex (business) logic in component, that logic or code base we cannot access in another template or html page.

To overcome this service concept, we use.

View/HTML Component Service File

HTML1 component ts

HTML2 component ts Service.ts

HTML3 component ts

Angular Service mainly divided into two types.

1. User-defined service
   1. Creating user defined service class object explicitly
   2. Creating user defined service class object using DI.

IOC : Inversion of control is a concept

DI : DI is a implementation of IOC.

Angular support only one type of DI ie constructor base DI.

If angular want to create the object for service class we have to follow few rules.

Imp Points to remember:-

We have to make class with decorator @Injectable

Then we must register this class details inside a module or component with property as provider.

Open app.module.ts file and provide the service class name inside a provider attribute.

Then in a component file we must pull the object using constructor.

1. Pre-defined service