**Phase 1**

**Day 1**

**02-08-2021**

**Web Technologies**

**https://**[**www.google.com**](http://www.google.com) **– URL : Uniform Resource Locator**

**http : protocol : hypertext transfer protocol.**

**Set of rules**

**https: secure**

**stateless protocol**

[**https://www.google.com**](https://www.google.com) **-🡪 URL**

**req(http/https)---🡪**

**Client Server**

**🡨--res(http/https)--- HTML/HTML5**

**HTML : Hyper text mark up language.**

**Static web page.**

**Dynamic web page.**

**HTML provide lot of pre-defined tags which help to create web page.**

1. **Html**
2. **Head**
3. **Body**
4. **Title**
5. **P**
6. **Heading tag h1 to h6**
7. **A anchor tag : hyper reference**
8. **Images**
9. **List tag**
10. **Table tag**
11. **Forms tag**

**In html 4**

**JSF : Java Server Faces .jsf**

**<!doctype HTML PUBLIC URL=”uRLAddress.dtd”/>**

**Document type definition**

**Root tag name ie html, two child head and body, inside head title, script, style and meta etc.**

**Inside body optional or more than one p, h1 to h6, div, span, table, form tags.**

**In html5 they remove this concept.**

**<!doctype html> : in html5 we are writing this tag giving the instruction to browser we are using html5 features(optional).**

**Before HTML5**

<input type="text/password/radio/checkbox/button/file/submit/reset"/>

**After HTML5**

<input type="number/email/date/url/color/"/> etc

**By default every html form method is consider as GET.**

**If method GET then Information send through URL using query param concept.**

**URL?key=value&key=value&key=value**

**In get method data not secure.**

**Method = post . if method is post data send through body part of request.**

**Div is know as division tag. It is also known as container tags. which can contains more than other tag like p, b, i, h1 to h6 as well as another div tag. Which help to represent the part of the web page.**

**HTML5 provided semantics tags.**

**CSS**

**Class and id selector**

**Span**

**Font and text properties diff**

**Box model 1**

**Flexbox 2**

**Grid model 3**

**Position**

**Float**

**background**

**Absolute and relative**

**@media**

**Frameset : deprecated iframe**

**Types of CSS**

1. **Inline**
2. **Internal css or embedded css**
3. **External**

**class : groups of tags. two tags can contains same class name.**

**id : to make tag unique we have to provide the id. Id must be unique.**

**<div>**

**<p class=”abc” id=”p1”>First Para</p>**

**<p class=”xyz abc” id=”p2”>Second Para</p>**

**<p class=”abc” id=”p3”>Third Para</p>**

**<p class=”xyz” id=”p1”>Fourth Para</p>**

**</div>**

**Box model**

**Border properties**

**Box Model : row or columns**

**Flex model : we can display the tag or dom elements may be row wise columns wise.**

**In CSS we have to write the style rules from a scratch.**

**Bootstrap : bootstrap is a external CSS open source framework. Which provide lot of pre-defined css classes with respective all tags like p, div, button, form, table etc.**

**First framework provide to make responsive web page or application.**

**Phase 1**

**Day 2**

**03-08-2021**

**Position : fixed, static, relative and absolute etc**

**By default position is static.**

**Bootstrap**

**We can add bootstrap features to our web pages**

1. **Using CDN URL**
2. **Download Bootstrap files.**
3. **Using Node JS.**

**Container and container-fluid**

**Container : container is pre-defined bootstrap class which is known as fixed width container.**

**Container –fluid 🡪 It is a pre-defined class which takes full width of the web page.**

**Bootstrap gridlayout**

**Grid layout is use to arrange the dom elements in row and columns format.**

**By default every two divided into 12 columns.**

**Screen size**

**Xs : extra small <576px**

**Sm : small >=576px**

**Md : medium >=768px**

**Lg : large >=992px**

**Xl : extra large >=1200px**

**JavaScript**

**=== vs ==**

**Control flow**

**looping**

**Alert and prompt**

**Anonymous function**

**Module and**

**Generic function**

**Array object**

**JavaScript Object**

**DOM**

**Set and map collection**

**Ajax**

**Promise**

**Asyn and await**

**JavaScript was object based interpreter scripting language Till ES5**

**ECMA : European Computer Manufacture Association**

**ECMA is a concept.**

**One of the implementation of ECMA (ES) is JavaScript.**

**Object based Vs Object oriented : no class keyword it contains only pre-defined object as well as we can create user-defined objects. Even we can call as protocol base.**

**Interpreter Vs Compiler : interpreter check the code line by line. Compile convert all code at time.**

**Using JavaScript we can do programming on web page.**

**HTML -🡪 content display**

**CSS/bootstrap 🡪 presentation on content or look and feel.**

**JavaScript 🡪 action on contents or programming on contents.**

**In JavaScript to declare the variable we use var keyword till ES5.**

**Control flow**

**If statement**

**if(condition) {**

**}**

**If else**

**if(condition){**

**}else {**

**}**

**If else if**

**if(condition){}**

**else if(condition){}**

**else {}**

**Switch statement**

**Looping :**

**While loop**

**Do while loop**

**For loop**

**For in**

**For of**

**forEach : it is a function.**

**Function and events**

**Function : function is use to write set of instruction to perform a specific task.**

**Pre-defined function**

**alert()**

**prompt()**

**parseInt()**

**parseFloat()**

**eval()**

**User-defined function**

**In JavaScript we can declare function in different ways.**

1. **Normal function declaration syntax**

**function functionName() {**

**}**

1. **Expression style function**

**var functionName = function() {**

**}**

1. **Arrow function : arrow function is a short cut of expression style function.**

1. **Callback : passing the function name or function body or function itself to another function is known as callback function.**

**Apply(), bind(), call() : ES5 style object creation.**

**Phase 1**

**Day 3**

**04-08-2021**

**In JavaScript we can create user defined object**

1. **Function style using ES5 style**
2. **Literal style using es5 style**
3. **Class style using ES6 style**

**array : array is use to store more than one value of same or different types.**

**syntax to create the array**

**literal style**

**var obj1 = [];**

**var obj3 = [10,20,30,40,50,60]**

**object creation style**

**var obj2 = new Array();**

**var obj4 = new Array(10,20,30,40,50,60);**

**Set : set is a type of data structure which help to store unique elements values.**

**Map : map is a type data structure which help to store the value in key value pairs.**

**Key is unique and value may be duplicate.**

**DOM : document object model**

**Index.html**

**<html>**

**<head>**

**<title>This is simple web page</title>**

**</head>**

**<body>**

**<p>Welcome to My Web Page</p>**

**</body>**

**</html>**

**Run application in browser internally DOM hierarchy create.**

**Html**

**Head body**

**Title p**

**textNode textNode**

**textData textData**

**If we want to read, write and update HTML contents dynamically. All programming language provide DOM API (Document object model application programming language).**

**Java, Python, C++, .net and JavaScript etc.**

**Ajax**

**Promise**

**Asyn and await**

**Synchronous and asynchronous operation**

**Synchronous statement execution : it execute line by line**

**document.write(“1st statement”);**

**document.write(“2nd statement”);**

**document.write(“3rd statement”);**

**Synchronous statement execution : it execute line by line**

**document.write(“1st statement”);**

**asyn document.write(“2nd statement”); it execute independently**

**document.write(“3rd statement”);**

**synchronous function call**

**fun1();**

**fun2();**

**fun3();**

**asynchronous function call**

**fun1();**

**asyn fun2();**

**fun3();**

**synchronous communication**

**1st req**

**2nd req**

**3rd req**

**Client Server**

**asynchronous communication**

**1st req**

**asy 2nd req**

**asy 3rd req both request work independently.**

**Client Server**

**setTime**

**setInteval**

**clearInterval**

**Ajax**

**Promise**

**Asyn and await**

**AJAX : Asynchronous JavaScript and XML**

**XMLHttpRequest : It is pre-defined object which help to do asynchronous communication.**

**Promise : Promise is a pre-defined object which help to handle asynchronous events of data. Data can be text, number, json or xml.**

**Promise can be resolve(successfully done) or reject(failure).**

**If we want to load the data from promise object**

**We have to use then() and catch()**

**Then will execute when promise resolve**

**Catch will execute when promise rejected.**

**Fetch() : it is pre-defined function which internally use ajax to call backend technologies services.**

**REST API call.**

**Backend technologies in Java, python, .net, node etc**

**When we call they return in text format or json format, xml format.**

**Fetch() function return type is promise. Using then we can load the data if promise resolve or if any error generate we can handle using catch.**

**Asyn and await**

**ES7 features : It is a replacement of then and catch.**

**User defined object creation**

**Oops**

**Object 🡪 any real world entity**

**Properties -- have – fields / variables**

**Person**

**Behaviour -- do/does – functions / methods**

**Place**

**Bank**

**Animal**

**Car**

**Object is concept.**

**//ES5 style object creation : function or literal style**

**//ES6 style object creation : class**

**var, let and const**

**using var keyword we can re-declare same variable once again. But using let keyword we can’t declare same variable once again.**

**Using var keyword we can create global scope or function scope. Let using let we can create block or local scope.**

**const a=10;**

**//a=20; we can’t change the value**

**Node JS**

**Overview of TypeScript**

**Angular Framework**

**Day 4**

**HTML,CSS,JavaScript and bootstrap**

**Library and framework**

**jQuery , backbone js, coffee js, angular js and angular framework, react js and vue JS etc.**

**Node JS : Node js is not a library or framework. It is run time environment for JavaScript library or framework.**

**Like a jre in java.**

**Before Node js JavaScript is known as client side scripting language. But after node js JavaScript we be use client side as well as serve scripting language.**

**Node js contains lot of pre-defined modules may be local module or external module which help to do server side programming language.**

**Using Node js we can do file handling programming, we can create server side program, we can create REST API, we can connect database like SQL or Non SQL database etc.**

**MEAN Stack : Mongo DB, Express Module, Angular Node JS**

**MERN Stack Mongo DB, Express Module, React Node JS**

**In node js we can’t use window and document objects.**

**BOM and DOM**

**Browser Object Model**

**Document object Model**

**Node js provide**

**npm (node package manager).**

**Npm is use to install external module.**

**Syntax to install external module**

**Npm install –g moduleName globally**

**Or**

**Npm install moduleName locally**

**Typescript : Typescript is a super set of JavaScript which support all ES6 features.**

**Angular JS : HTML,CSS,JavaScript using ES5 or ES6.**

**1.x 1.8 etc**

**MVC base architecture**

**Angular Framework 2 to 12 : technologies require**

**Html, css, Typescript.**

**Component base architecture**

**Browser can’t understand Typescript scripting language.**

**So we have to convert ts to js.**

**Traspiler ( it is type of compile which help to convert ts to js).**

**tsc (typescript compiler).**

**To convert ts to js we require tsc**

**npm install –g typescript**

**Typescript features**

**Typescript support data types.**

**number, Boolean, string, any**

**we can use array with data types.**

**TypeScript function**

**Typescript OOPS concept.**

**Using import and export we can access function, classes, interfaces from more than one file in typescript.**

**Modules : it is collection of function, variable, classes and interfaces.**

**In typescript two type of modules**

**Internal modules**

**External module (angular use).**

**We have to create typescript configuration file**

**tsconfig.json**

**tsc –-init : this command is use to create tsconfig file.**

**Angular project creation**

**First install node js**

**Using node js command install ng**

**npm install –g @angular/cli**

**(next generation)**

**ng new project-name**

**routing 🡪 no**

**styling -> css**

**after project created successfully**

**move inside a project directory**

**cd project-name**

**ng serve : to run the project**

**after compiled successfully**

**open browser** [**http://localhost:4200**](http://localhost:4200)

**decorator**

**property binding and string interpolation**

**forms**

**Observable**

**dependency injection and service**

**routing : linking pages, wild cards.**

**Angular is a open source framework provided by google.**

**Angular is use to create SPA Singe Page application.**

**Multi page application.**

**Index.html welcome.html**

**Hyperlink**

**Button**

**Submit button**

**Etc**

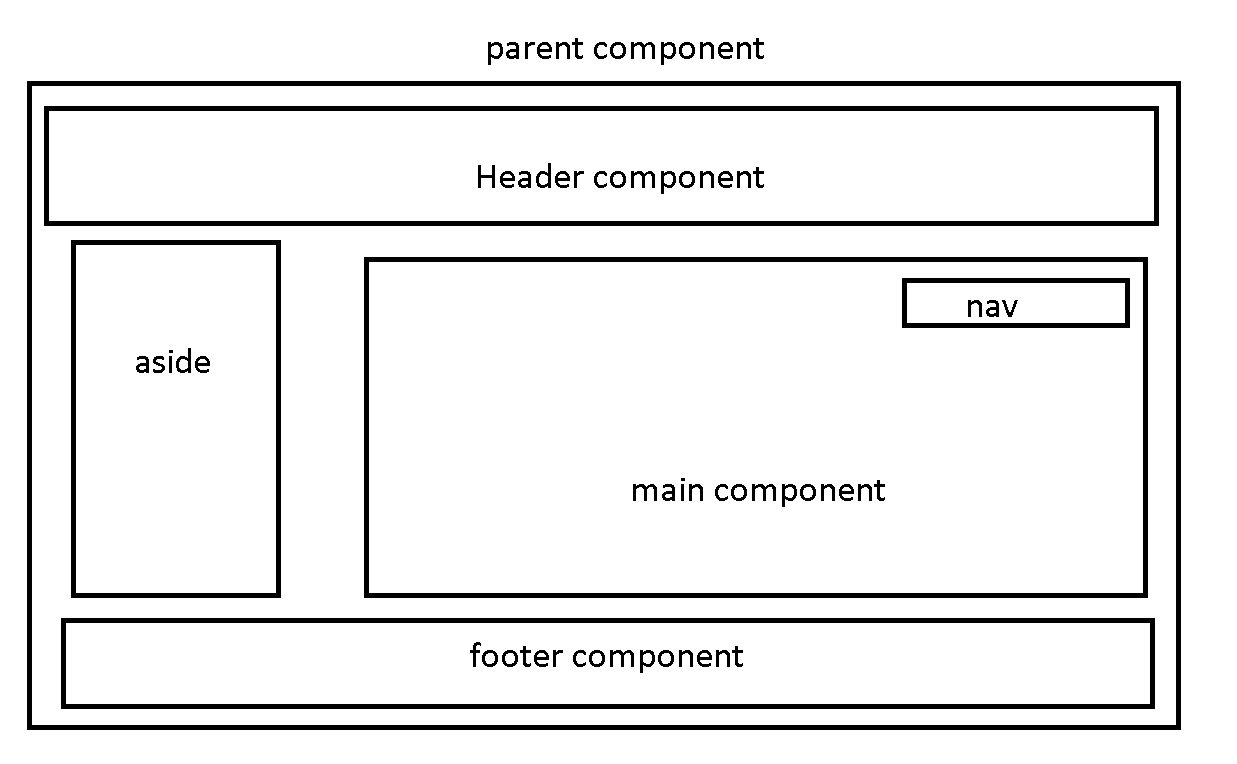
**Single page application :**

**Rather than loading whole page we can’t load the part of the in first page.**

**Component : it control the view or part of the view.**

**Module is a combination of more than one components.**

**Module is a like a package in java.**

****

**Src**

**App**

**Open app.component.html**

**Write your own html code**

**No html, head, and body tags.**

@Component

**Typescript provide one of the great features ie**

**Decorator : decorator is a type of special function which help to make class or property as special type. Decorator is also known as meta data (data about data). Decorator like a annotation in Java.**

**All decorator start with @ symbol and followed by decorator name.**

**@Component**

**@NgModule**

**@Injectable**

**@Pipe**

**@Input**

**@Output**

**@ViewChild**

**Etc**

**@Component decorator contains lot of property**

**selector : unique name for that class**

**templateUrl**

**styleUrl**

**in Angular using component we are creating user-defined tags**

**selector is behave like a user-defined tags.**

**templateUrl : connected to html page**

**stylesUrl : connected to css page**

**app.module.ts**

@NgModule: module is a collection of more than one components.

**declaration : all component declaration details provide in this section.**

**Imports : this property is use to import pre-define or user defined modules.**

BrowserModule: this module help to use to display or render the data in browsing area.

**Provider : inside this property we will provide the angular service class details.**

**Bootstrap : this property is use to load the parent components.**

**nain.ts**

**This file load the main or parent modules.**

**appModule LoginModule CustomerModule OrderModule**

**appComponent createLoginComponent 🡪 one html page.**

**displayLoginComponent**

**deleteLogincomponent**

**angular.json file**

"index": "src/index.html",

"main": "src/main.ts",

**Open index.html**

**<app-root></app-root>**

**package.json file**

**this file contains angular version and other dependencies details.**

**Index.html welcome.html**

**<html>**

**<body>**

**<a href=”welcome.html”>click</a>**

**</body>**

**</html>**

**Whole dom once again loaded in browser.**

**<html>**

**<body>**

**<h2>WElcome to Home Page </h2>**

**</body>**

**</html>**

**index.jsp**

**<html>**

**<body>**

**Header jsp footer jsp nav jsp**

**<h2>WElcome to Home Page </h2>**

**</body>**

**</html>**

**Header.jsp : it own dom**

**Footer.jsp : it own dom**

**Nav.jsp : it own dom**

**Index.html**

**Header html include**

**<footer>**

**Coding….**

**</footer>**

**Dashboard.html**

**Include header html**

**<footer>**

**Coding….**

**</footer>**

**Header.html**

**<header>**

**Coding…..**

**</header>**

**3 to 4 project**

**ng new angular-data-binding**

**ng new angular-forms**

**ng new angular-service**

**ng new angular-routing (routing yes)**

**we can create the component using ng command**

**ng generate component componentname**

**or**

**ng g c componentname**

**ng serve –o : after compiled program automatically open in default browser.**

**data binding : data binding provide the bridge between html (template) and component. Data binding help use to share the data between template to component.**

1. **One way data binding**
2. **String interpolation : component ----🡪 view**

**{{variableName}}**

**{{expression}} {{5+6}}**

**{{functionName()}}**

1. **Property binding : component ----🡪 View**

**[]**

1. **Event binding : view --🡪 Component**

**() with on pre-fix. Only event name must wrap with ().**

**Angular using same dom events**

**DOM JavaScript event Angular Event**

**onClick (click)**

**onDblClick (dblclick)**

**onMouseOver**

**onMouseOut**

**using event binding and string interpolation /property binding we can do 2 way data binding**

**passing the value from component to view.**

**Template Reference**

1. **Two way data binding**

**Angular support 2 way data binding**

**[()]**

**To achieve the two way data binding angular provide ngModel pre-defined attribute. ngModel pre-defined attribute part of FormsModule. So we have to import this module in app.module.ts file.**

**Angular Forms**

**Angular support two type of forms**

1. **Template Driven Form : view 🡪 component**

**Easy to develop. Good for develop small application**

1. **Model Driven form or reactive forms: component 🡪 view**

**Complex we have to write more code in ts file. Good for complex forms.**

**Template Driven Form**

**In template driven form we have to create the form reference**

**<form #loginRef=”ngForm”>**

**</form>**

**ngForm is a pre-defined attribute part of FormsModules so we have to import FormsModule in app.module.ts in import section.**

**Name :<input type=”text” #nameRef/>**

**Password :<input type=”password” #passRef/>**

**Model Driven Form**

**According to model driven form or reactive forms**

**Every html form component is known as FormControl**

**Like textField, passwordField, radiobutton, checkbox, dropdown etc.**

**FormControl must be wrap inside a FormGroup.**

**Login Page -🡪 FormGroup**

**TextField🡪FormControl**

**Password 🡪FormControl**

**FormGroup contains one or more form control as well as another form group.**

**Angular provide FormGroup and FormControl pre-defined API to bind with Reactive forms.**

**In component we have to create FormGroup reference and which can contains more than one FormControl reference as well as FromControl.**

**Those reference we have to bind with in html using formGroup and formControlName attribute.**

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

**If we write the business logic in component it may be simple or complex that logic become local to that component.**

**But if we want to some business logic globally accessible in all component then we can use service class.**

**Service**

**Component1 fun1()**

**Component2 fun2() fun()**

**Component3 fun3();**

**Angular Service divided into two types**

1. **User-defined service**
2. **Creating service class object using new keyword**
3. **Creating service class object using DI**
4. **Pre-defined service**

**IOC : Inversion of control : rather than creating or maintain any resource explicitly allow to maintain and create by container. If container create it maintain properly. Design patter or programming pattern.**

**IOC is concept.**

**DI : Dependency Injection**

**Implementation of IOC is DI.**

**Constructor base**

**Setter base**

**Interface base**

**But angular support only one type of DI is constructor base.**

**If we want to make DI for service class we have to create normal type script class with @Injectable decorator.**

**Then provide the details about service class in app.module.js file in provide section.**

**Angular provide pre-defined HttpClient API which help to call REST API or load json Data.**

**So in service class we have to do the DI for HttpClient.**

**HttpClient is a predefined API part of HttpClientModule so we have to import HttpClientModule in app.module.ts file.**