Day 1 : 05-03-2021

Git :

Application Or Project

A Person Login module

B Person Customer module

C Person Account module

Merge code

Version Control tool

Git : Git is a version control system for tracking changes in file or folder or application or project and coordinates work of those file or projects among the multiple people.

Create folder : then create one or more than one file with simple message.

Git init : to create local repository

Git status :

Git add filename.extension

Git add a1.txt a2.txt a3.txt

Git add \*.txt

Git add .

Git commit –m “File created”

Git status

Git init

Then create the file

Git add .

Git commit –m “message”

Git remote add origin URL

Git push –u origin HEAD

Or

Git push – origin master

Day 2 : 08-03-2021

Git init : This command is use to create local repository.

Git status : This command is use to check the last command status.

Git add . : This command is use to add file or folder or project in staging area.

Git commit –m “Message” : This command is use to pass the files or folder from staging area to local repository.

Git remote add origin URL : This command is use to link local repository to remote repository.

Git push –u origin HEAD/branchName: This command use to pass the value from local repository to remote repository.

Git clone URL : This command is use to download the remote repository to local machine.

Manager 🡪 Simple.txt

File created by Manager

Git init

Git add .

Git commit –m “File created”

Git remote add origin URL

Git push –u origin HEAD/BranchName

Default branchName –master/main

Raj

Git clone URL

Master/main

Ajay

Git clone URL

Master/Main

Branch : Branch is like a pointer which hold more than one commit details.

Git branch : This command is use to check the all branch details.

Git branch branchName : This command is use to create the branch.

Git checkout branchName : This command is use to switch from one branch to anther branch.

Git checkout –b branchName: This command is use to create the branch and switch to created branch.

Git branch –D branchName : This command is use to delete the branch locally.

Manager -- > Create main file

And push to remote repository

Raj Developer -🡪 clone remote repository

Create the file in Raj branch

Push raj branch to remote repository

Ajay Developer 🡪 Clone remote repository

Create the file in Ajay branch

Push ajay branch to remote repository

Create Folder TCSMEANStackTraining

Test file

Your Details

Remote Repository : EmpId\_YourName\_TCSMEANStackTraining

Web Technologies.

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

http/https🡪Req---🡪

Client Server

🡨-res(http/https)---

http/https: Hyper text transfer protocol.

Protocol : set of rules which help to communicate more than one machine or device.

www: world wide web

google : domain or search engine.

com : commercial

Uniform Resource Locator

HTML : Hyper Text Mark up Language.

It is use to display the contents.

CSS : Cascading Style sheet

Look and Feel or Presentation on Contents.

JS : JavaScript

Action on Contents

HTML : Hyper Text Mark Up Language : it is use to create the web page.

Web Page : It is use to display the contents in different format like normal, bold, italics, video, audio, clips etc.

Web Application : Collection of more than one web page.

Using HTML we can create static as well as dynamic web page.

Static : Display contents as it is on browser.

Dynamic : When user interact with web page event(action performed) generate.

Version

1, 2, 3, 4 and 5 : HTML5

Tags or elements.

Syntax

<tagName> opening tag

</tagName> closing tag

<tagName/> self closing tag

HTML is case insensitive.

HTML tags

Html

Head

Body

Title : This tag is use to display the message in title bar.

Paragraph tag : This tag is use to display the contents in browsing area.

This tag must in in between body tag.

<p> </p>

Break or br tag : This tag is use to break the content in next line. Break tag doesn’t contains closing tag.

<br>

<br/>

Heading tag : This tag is use to write the heading for paragraph or any contents.

There totally 6 heading tag

H1 : largest

To

H6 : smallest

Day 3 : 09-03-2021

Attribute : attribute is know as properties of a tags.

Syntax

<tagName name1=”value1” name2=’valule2’ name3=value3></tagName>

Attribute must in opening tag

In the form of key-value pairs.

Key may in single quote or double quote or without quote.

If value may be more than one world then it must be in single or double quote.

<p align=”center”></p>

<h1 align=right></h1>

Hyperlink :

This tag is use to connect external as well as internal (bookmark) contents.

External hyper link

Internal hyper link (book mark)

External hyper link

<a href=”pathOfFile.html”></a>

a 🡪 anchor

href 🡪 hyper reference.

Hr : horizontal line

<hr/>

Internal hyper link or book mark

<a href=”#a1”></a>

<a href=”#a2”></a>

<a name=”a1”></a>

<a name=”a2”></a>

Image tag

<img src=”NameOfImage.extension”/>

List Tag :

Unorder List

<ul>

<li>Raj</li>

<li>Ravi</li>

<li>Ramesh</li>

</ul>

UL : Unorder List

Li : List item

Order list

<ol>

<li>Java</li>

<li>Python></li>

<li>JavaScript</li>

</ol>

Ol : order list

Li : list item

Definition list

DL : definition list

DT : Definition term

DD : definition description

Table Tag :

Employee Details

Id Name Salary

100 Raj 12000

101 Ramesh 14000

<table>

<tr>

<th>Id</th>

<th>Name</th>

<th>Salary</th>

</tr>

<tr>

<td>1</td>

<td>Ravi</td>

<td>12000</td>

</tr>

</table>

Tr 🡪 table row

Th 🡪 table heading

Td 🡪 table data

Form Tag

Login, Application, Feedback Page, etc

<input type=”text/password/radio/check/button/submit/reset/file” />

By default HTML form method is Get consider.

If method is get information send through URL using URL rewrite technique.

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

Get is not a secure.

If you want secure then use method is POST

Get faster than post method.

VS Code :

Day 4 : 10-03-2021

HTML 4 version

<html public =”URL .dtd”>

<head>

</head>

<body>

</body>

</html>

Document type definition

HTML 5

<!doctype HTML> : Given the instruction to browser writing HTML5 version features.

VS Code

: Visual Studio Code :

MEAN Stack

Mongo DB Express Angular Node JS

CSS :

If we want to apply any formatting style for the contents we have to depending upon other tags.

As well HTML provided only few tags with the help of those tags we can’t apply good look and feel for web page.

Using HTML if we apply formatting style.

The content and formatting style combined together.

CSS : Cascading style sheet. CSS provide lot property in the form of key-value pairs which help to apply good look and feel for the web page.

With help of CSS we can achieve separation of concern.

Actual Content Formatting style

CSS file divided into 3 types

Inline css

Internal or Embedded CSS

External CSS

Inline CSS

Syntax

<tagName style=”property:value;property:value;”>

</tagName>

P,h1,h6,b,form,input,table, tr, td etc

Internal or Embedded CSS

In between head tag we have to write style tag

<style type=”text/css”>

Selector {property : value;property:value}

</style>

Types of selectors

Universal selector : \*

 \* {color: skyblue;}

Specific selector

tagName {property : value}

p,h1,h6,div,span, b etc

Multi specific selector

tagName,tagName,tagName {property:value;}

Local Class selector :

tagName.className{property:value}

Global class selector

.className{property:vaulu;}

Id selector :

#idName {property:value;}

Class selector Vs Id selector

More than one tag must be part of same class

Or

Class is a group of more than one tags.

Id is use to give unique ness for the tags.

<p class=”abc” id=”a1”>First Para</p>

<p class=”xyz” id=”a2”>Second Para</p>

<p class=”abc” id=”a3”>Third Para</p>

<p class=”xyz” id=”a4”>Fourth Para</p>

Div tag : Div tag is known as container tag. Which contains more than one other tags like p, h1, span, as well as another div.

External CSS File : CSS rules globally available for all web pages.

Font related property

Day 5 : 11-03-2021

CSS3 Property

CSS3 Transformation : Transformation allow you to translate, rotate, scale and skew html element (DOM (Document Object Model)).

CSS3 Transition : CSS3 Transition allow you to change CSS property value smoothly over a period of a time.

CSS property you want to add an effect to like width, height, color, size etc

Duration (time).

CSS3 Animation :

CSS3 animation allow most of HTML tags with JavaScript or Flash.

@keyframes : Which contains set of rules to execute the animation (ie start and end position for animation).

JavaScript

JavaScript was object based interpreter scripting language.

ES5 ECMA Script : European Computer Manufacture Association.

Using JavaScript we can do coding or programming on web page.

JavaScript tags

Syntax

<script type=”text/JavaScript”> opening tag

</script> closing tag

This script tag we can write in between head tag or body tag or without any tags.

In one html page we can write more than one script tags.

Variable and data types.

Variable is a name which hold the value.

In JavaScript to declare the variable we have to use the keyword as var(up to ES5 JavaScript).

var variableName;

Data types : Data type is a type of data which tells that type of value it can hold.

JavaScript data types

Number (with or without decimal)

String : more than one character single or double quote.

Boolean : true or false.

Object reference.

Operator

1: Arithmetic Operator : +, -, \*, /, %

2 Relational operator >, >=, <, <=, ==, !=

Assignment operator : =

=== :

Logical operator : &&, ||, !

Typeof operator or function

If statement

Simple if

if(condition) {

}

if else

if(condition) {

}else {

}

if else if

if(condition){

}else if(condition) {

}else if(condition) {

}else {

}

Switch statement

Syntax

switch(variableName) {

case label1: block1;

break;

case label2: block2

break;

case lable3:block3

break;

default : defaultblock

break;

}

Avg =80

if(avg>90) {

a++

}else if(avg>80) {

a

}else if(avg>70) {

b

}else {

c

}

Looping : It use to execute the statement again and again till the condition become false.

While loop

Do while loop

For loop

Initialization : start / end

Conditions true

Increment / decrement

styles.css (external CSS)

.className : global class selector

tagName.className : local class selector

.divFontClass{

font-size:

font-style:

font-family:

}

<div class=”divFontClass”>

</div>

<p></p>

<input type=”button”/>

<h1></h1>

<table>

Min 3 tag for 3 class selector (CSS class selector which contains min 3 css property).

Function and events

Function : it use to write set of instruction to perform a specific task.

2 types

pre-defined function

document.write(“Welcome to JS”);

alert(“welcome to JS”);

prompt(“Msg”): This function is use to receive the value from keyboards.

eval() : This function is use to convert string to number.

parseInt(): string to int

parseFloat(): string to float

eval() = parseInt() + parseFloat()

user-defined function

syntax

function functionName(parameterList) {

}

Event : Event is a interaction between user and html tags ( components) or DOM (Document Object Model).

DOM : All html tag is known as DOM elements. Like html, p, h1, b, form etc.

Event provide the bridge between html and JS code.

Type of events

In JavaScript all event start with prefix on followed by event name

onClick

onDblClick

onMouseOver

onMouseOut

onKeyUp

onKeyDown

onSubmit

onChange

onBlur

onFocus

etc

External JavaScript file

JSON : Java Script Object Notation

JSON is use to share the data between one application to anther application.

JSON is use to store the data in the form of key-value pairs. Where key is string type and value may be number, string, boolean, object reference type.

Object / JSON to String conversation

JSON.stringify()

String to JSON conversation

JSON.parse()

Storage objects

HTML5/JavaScript provide two types of storage object

sessionStorage : It hold the value till application close. Once application close the value get destroy from session.

localStorage : It hold the value in secondary memory after close the application also we can get next time open the application.

Using storage object we can share the value between more than one page or application.

One.js

sessionStorage.setItem(“key”,value);

localStorage.setItem(“key”,value);

two.js

sessionStorage.getItem(“key”)

localStorage.getItem(“key”);

sessionStorage.removeItem(“key”);

localStorage.removeItem(“key”);

**Day 7 : 15-03-2021**

**Types of function in JS**

1. Normal function
2. Expression style function
3. Callback function : passing the function body or function itself to another function is known as callback functions.
4. Arrow function
5. Using Array with normal function, anonymous function and arrow function
6. IIEF function.

**Creating User defined objects Using ES5 style**

function Employee() {

}

Synchronous and Asynchronous Communication.

Synchronous Communication : This code is executed in sequence.

Asynchronous communication : This code execute independently.

setTimeout() : it is pre-defined function which is use to achieve asynchronous communication.

**Promise :** Promise is a pre-defined object provided by JavaScript which help to handle asynchronous operation.

It also handle error.

4 has state

1. Fulfilled : When promise is success.
2. Rejected : when promise is failure
3. Pending : when promise is pending state.
4. Settled : when promise completed.

**Day 8 : 16-03-2021**

**Bootstrap :** It is a open source CSS framework which help for developing responsive and mobile – first application or websites.

.fontClass {

font-size:24pt;

}

<div class=”fontClass”>

Information

</div>

**Bootstrap pre-defined css classes**

**Container : Container is the fixed width container. It is fixed base on the screen size.**

**Container-fluid : This type of class take the full width of the device.**

**Bootstrap grid layout :**

Boot strap grid layout use a series of rows and columns to layout and align the contents.

In Grid layout each row divided into 12 columns.

**Navigation** is type of bootstrap css class using **.nav** class to activate and disable states.

**Phase 2 :**

**Day 9 : 18-03-2021**

**Node JS : Node JS is a run time environment for Java Script library and Java Framework.**

**Java – JRE**

**JavaScript – Node JS**

**To do typescript program as well as Angular program it require node js.**

**TypeScript**

**Angular 9/10/11**

**Typescript : Typescript is super set of JavaScript. Typescript support ES6 as well as ES7 all features.**

**Features of TypeScript**

1. **To declare variable var, let and const**
2. **Support all data types. number, string, Boolean, Array, generics etc.**
3. **Different type of functions**
4. **Different type of loop** 
   1. **Of loop and in loop**
5. **OOPs concept using class, interface, constructor, static keyword,**
6. **Modules.**

**To run simple external JS file we were using HTML page.**

**<script src=”demo.js”></script>**

**With help of node Js we can run JavaScript program without html page.**

**Using command prompt we can run external JavaScript programs.**

**If we are planning to run javascript program through node js. In that program don’t use window as well as document.**

**First create**

**demo.js**

**console.log(“node js program”);**

**.js**

**If file is .js we can run through html page or node js.**

**.ts (typescript).**

**If file is ts file we have to convert this ts file into js file.**

**It require transpiler (like interpreter) which help to convert ts to js.**

**Tsc (typescript compiler) : this command help to convert ts to js.**

**npm (node package manager) : npm is use to download external modules.**

**Command to download external modules**

**Syntax**

**npm install –g moduleName**

**npm install –g typescript**

**or**

**tsc**

**var,let and const**

1. **var : var keyword is use to declare the variable. It is use to declare the global scope.**

**Using var keyword we can declare same variable with same or different values.**

1. **let : let keyword is use to declare the local or block scope.**

**Using let keyword we can’t to re-declaration.**

**Typescript support data types.**

var variableName:datatype;

let variableName:datatype;

**Array declaration :**

Up ES5

ES5 array declaration

var num=[100,200,300,”Ravi”,true];

ES6 using TypeScript

var num:number[]=[100,200,300,400,500];

different type of loop

for in loop

for of loop

**Array methods**

Till ES5

function functionName(a,b,c){

}

functionName(10);

functionName(10,20);

functionName(10,20,30);

functionName(10,20,30,40);

functionName(“Ravi”);

**Phase 2 :**

**Day 10 : 19-03-2021**

**OOPs Concept Using Typescript**

**object : any real world entity.**

**State / properties – have --fields/variables**

**Person   
 behaviour ---do/does –function / methods**

**Bank**

**Animal**

**Car**

**Employee**

Employee class ES6 style

class Employee {

    id:number=100;

    name:string="Ravi Kumar";

    salary:number=12000;

    empInfo():void {

        console.log("id is "+this.id);

        console.log("name is "+this.name);

        console.log("salary is "+this.salary);

    }

}

let emp1 = new Employee();

emp1.empInfo();

let emp2 = new Employee();

emp2.empInfo();

**constructor : constructor is use to create the object or memory.**

**To create the constructor we have to write name for the function as constructor. Constructor get call automatically when we create the object.**

**Difference between constructor and instance functions.**

**In the life of the object if we want to perform any task only one time that type of code we have to write inside a constructor.**

**In the life of the object if we want to perform any**

**More than one time that type of code we have to write inside a functions.**

**In Typescript we can’t write more than one constructor may empty or parameterized.**

**Private property or function we can’t access outside class using object as well as directly.**

**Public we can access using object of that class.**

**Inheritance : It is use to inherits the properties and behaviour of old class to new class.**

**Interface :**

**Interface is use to write more than incomplete functions.**

**Syntax**

**interface interfaceName {**

**variables;**

**incomplete function**

**}**

**In typescript write interface which contains only variable or only function (incomplete).**

**Modules : modules is a collection of variable, function, classes , interfaces which have same name different purpose.**

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

**Using module we can avoid the conflicts when two function or classes or interface have same name but different purpose.**

**Phase 2 :**

**Day 11 : 22-03-2021**

**HTML/HTML5**

**CSS/CSS3**

**JavaScript ES5 – ES6**

**Bootstrap**

**TypeScript**

**DOM Document Object Model**

**Read, Write and Update DOM Elements (HTML Tags) dynamically.**

**jQuery : jQuery is a external JavaScript library. Which contains set of functions which internally connected to each other. Which help to Read, Write and Update DOM very easily.**

**Library : Library doesn’t follow standards rules and regulations.**

**Framework : Framework follow standards.**

**Framework is know as Prototype or template but not final product. If we develop any application using framework 70% to 80% task taken are by framework.**

**The implementation of design pattern is taken care by framework.**

**Angular : Angular is a open source web framework.**

**Angular JS : HTML/CSS/JavaScript using ES5**

**Angular Framework 2 to 11. : HTML/CSS/TypeScript (JavaScript).**

**Angular Framework help to develop SPA (Single Page Application).**

**Multi page application.**

**Index.html home.html**

**Hyperlink**

**Button (submit)**

**Command button**

**Angular Framework use component to create the application.**

**Component use to control the view or part of the view on web page.**

**In angular every component work independently.**

**One component can interact with another components.**

**React JS is library . Library is not a standard**

**React JS + Redux or Flux (state management tools).**

**Using Angular framework we can develop standalone(desktop), web application as well as mobile application.**

**Angular framework is a part google company.**

**Angular JS: it is use only for web application.**

**Angular CLI (Command Line Interface). So with the help of Angular CLI we can create angular project very easily.**

**Ng : next generation on HTML.**

**Angular**

**npm install –g @angular/cli**

**npm install –g @angular/cli@versionNumber**

**Step to create the Angular project using Angular CLI**

**ng new project-name**

**ng new welcome-app**

**For more information, see https://angular.io/strict Yes/No : select y or n**

**Routing : Y/N : select y or n**

**Style : CSS : css**

**And enter the key.**

**After project created successfully . move inside a project using cd command.**

**cd welcome-app**

**To run the project we have to run the command as**

**ng serve**

**or**

**if you want to change port number**

**ng serve –port=4300**

**after successfully complied 100% project**

**open browser**

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

**By default angular provide web server to run the project. The project run by default port number 4200.**

**Open the project in VSCode**

**Then expand**

**src -🡪 app🡪**

**app.component.html**

**(In angular we call as template page)**

**app.component.ts (Component file)**

import { Component } from '@angular/core';

**Component is a pre-defined API(Application Programming interface) which part of @angular/core;**

**import {dis1} from ‘./a’;**

**@NamesOfDecorator : It is know as a decorator. Decorator is concept of typescript. It is like a annotation in Java. Decorator provide meta-data. Data about data.**

**Using decorator we can add extra features to class or property.**

**@Component : It is a pre-defined decorator provided by angular framework. it is type of decorator which help to control the view or part of view.**

**Component decorator property**

**selector : “user-defined-tag-name”: using selector we are creating user-defined tags.**

**selector is use to create the user-defined tags.**

**<h1>Welcome to HTML page</h1>**

**<p>Welcome to HTML Page</p>**

templateUrl: './app.component.html',

**with the help of templateUrl we are connecting ts file to html page.**

styleUrls: ['./app.component.css']

this property which help to connect ts file to external css file. It is like a link tag in html with CSS.

**app.module.ts**

**module is a collection of more than one components. It is like package in java.**

@NgModule: it is a type of decorator provided by angular to make the class is type of module class.

declaration : we have to provide the details of all components.

import : it use to import pre-defined or user-defined modules.

**Provider : This property is use to provide angular service details.**

**bootstrap :**

**we can create more than one modules and each module contains more than one components.**

**Login Modules**

**LoginCreate, LoginDisplayComponent**

**Application Modules**

**ApplicationCreate,ApplicationDisplay**

**Customer Modules**

**main.ts :This file provide the details about main modules to load using bootstrap.**

**A Module B Module C Module**

**If A is main module then B and C are child module**

**A module**

**Abc component : Parent Component**

**Xyz component : child component**

**Mno component : child component.**

**AppModule 🡪 root module**

**AppComponent 🡪 parent components.**

**index.html**

**This tag contains root-tag details.**

**2 angular projects**

**ng new angular-data-binding**

**ng new angular-forms**

**please don’t create nested projects.**

**No routing options**

**Phase 2 :**

**Day 12 : 23-03-2021**

**Creating new component with local template and local CSS file.**

**ng serve :**

**ng serve –o : after loaded 100% successfully it automatically open in system default browser.**

**Creating new component using angular cli command**

**ng generate component componentname;**

**or**

**ng g c componetname**

**Data binding :**

**Data binding is use to share the data between template (html) to component (typescript file).**

**It provide bridge between template(view) to components.**

**View ---------------------------------Component**

**2 types**

1. **one way data binding** 
   1. **string interpolation : component --🡪view**

**{{variableName}}**

* 1. **property biding : component 🡪 view**

**[]**

* 1. **event binding : view---🡪 component**

**Angular use same event provided by JavaScript but remove on prefix and event name must wrap using ().**

**Javascript event Angular**

**onClick (click)**

**template reference :**

**syntax to create the template reference**

**#referenceName**

**Angular directive : angular directive is use to add extra behaviour for DOM Elements.**

**3 types**

1. **component directive :@Component it use to create the component directive. Which help to create the user-defined tags with the help of selector.**
2. **structure directive : Using structure directive we can add dom as well as remove dom(html tags) from web page.**

**\*ngIf : show and hide**

**\*ngFor : iterate collection of data.**

1. **attribute directive : it use to add styling for web page with help of ngStyle and ngClass.**
2. **two way data binding**

**angular-form**

**angular-routing :**

**while creating the project it ask touring options**

**yes**

**Day 13 : 24-03-2021**

**Angular forms :**

**Using angular we can create two types of forms.**

1. **Template Driven Form (TDF)**
2. **Reactive Form Or Model driven Form (MDF)**

**Template Driven Form**

**View(Template) --- > Component : HTML**

**It is very easily to create.**

**Good for simple forms.**

**Model Driven Form**

**Component --🡪 View (Template) : TypeScript**

**It is good for complex forms.**

**Create two component**

**ng g c tdf-login-page**

**ng g c mdf-login-page**

**In Template driven form we have to create the reference of form using ngForm attribute.**

**Syntax to create**

**In form opening tag**

**<form #userRef = “ngForm”>**

**</form>**

**Once you create the reference of form using ngForm in template page we will get the error.**

**Because ngForm is a part of FormsModule. So we have to import FormsModule in app.module.ts file**

**If we want to pass the textfield, passwordfield, radio button, checkbox etc value through form reference we have to write ngModel attribute in every html texfield, passwordfields etc.**

**Model Driven Form or Reactive Form**

**Component --🡪 View (Template)**

**Angular provided set of API(Application Programming interface) to achieve model driven form**

**FormGroup and FormControl**

**According to Model Driven form**

**TextField, PasswordField,RadioButton, CheckBox etc are consider as FormControl.**

**FormGroup contains one or more than FormControl as well as another FormGroup.**

**FormGroup and FormControl API is a part of ReactiveFormsModule. So we have to import ReactiveFormsModule in app.mdoule.ts file.**

**Angular Routing :**

**Angular routing is use to navigate from one component to another component base upon the path provided in routing file.**

**If path match one component’s template (HTML) code replace by another component’s template(HTML)**

**Aboutus component**

**Contactus component**

**Login component**

**Dashboard component**

**ng g c aboutus**

**ng g c contactus**

**ng g c login**

**ng g c dashboard**

<router-outlet></router-outlet>

**It is a pre-defined tag which behave as a placeholder to load the component’s template contents depending upon the path provided in router-module.**

**Day 14 : 24-03-2021**

**Angular – routing :**

**Auth guard**

**Auth guard : Auth guard is mechanism which help to restrict the user to redirect depending upon the conditions.**

**Angular provided different types of auth guards**

1. **CanActivate**
2. **CanDecative**
3. **CanActivateChild**

**Etc**

**All auth guard are interfaces.**

**Angular service**

**HttpClient Service**

**Angular Service : When we write any business logic in component it may simple or complex that logic become local to that component.**

**Using angular service we can provide any business logic (it may simple or complex) to all component like globally access.**

**Component1**

**Component2**

**Component3**

**Angular Service class**

**fun()**

**Using Angular service we can achieve separation of concern.**

**View -🡪 component -🡪 service**

**View ------Com-------🡪 Service**

**IOC : Inversion of control: in place of creating any resource or object explicitly allow to create and maintain by container(engine). IOC is known as Design pattern. It is concept.**

**DI : Dependency Injection : DI is a implementation of IOC. DI is use to pull the object or resource from a container whenever required.**

**3 ways**

**Constructor base**

**Setter base**

**Interface base**

**But Angular support only Constructor base DI.**

**Create user-defined class with @Injectable decorator**

**Then register class details in module level or component level with the help of provider attributes.**

**If we write user-defined class details in module level then it is consider a singleton object available for all component belong to that module.**

**Singleton mean only one object created which we can access in all components.**

**If we write user-defined class details in component level then it is available within that component.**

**Angular pre-defined service.**

**Angular provide pre-defined API to call backend service (REST API).**

**HttpClient is a pre-defined API which help to call REST API develop in any language like Node js, Java, .net, python, or php etc.**

**HttpClient provide pre-defined Http protocol methods like Get, post, put and delete etc.**

**So we have to do DI for HttpClient in user-defined service.**

**After DI done in FakeService class using constructor we will get the error.**

**Because HttpClient API is a part of HttpClientModule**

**So we have to import HttpClientModule in import section in app.module.ts file.**

**Day 15 : 24-03-2021**

**this.http.get(“URL”)**

**HttpClient get(), post(), put() and delete() methods return type is Observable.**

**Observable Vs Promise**

**.subscribe(next,error,completed);**

**This method takes three parameter as a callback functions. 1st parameter get call to load all data if REST service return success and load the data one by one, if any error generated then 2nd parameter called. If no error after 1st method after loaded successfully all data it call 3rd parameter.**

**1st parameter mandatory to load the data.**

**2nd and 3rd optional.**

ngOnInit(): void {

  }

**It is a life method it call only once when the component get loaded.**

**This method called after constructor.**

**According Angular constructor is use to do DI.**

**Don’t write any business logic or calling any service method inside a constructor.**

**To Load data from JSON File**

**Creating service class through command line**

**ng g s servicename**

**ng g s employee**

**ng g c employee-retrieve**

providedIn: 'root'    // it is equal to provided in app.module.ts file

**2-way data binding**

**One way data binding**

**{{}} : string interpolation**

**[]: property binding**

**() : event binding**

**2 way data binding**

**[(ngModel)]=”variableName”**

**If we do any changes in template it automatically updated on component and vice-versa.**

**Component 🡨-------------🡪 View**

**ngModel is a attribute part of FormsModule. So we have to import FormsModule in app.module.ts**

**Day 16 : 29-03-2021**

**ng new angular-pwa-service-worker**

**angular PWA (Progressive Web Application).**

**Service Worker**

**Json-server**

**PWA : Progressive Web Application that has a set of capabilities (similar to native apps) which provide an app like experience users. PWA need to meet a set of essential requirements.**

**PWA is like a bootstrap for the angular.**

**Angular materialization**

**If we want angular PWA features then we have to add using command as**

**Move inside a project using command**

**ng new angular-pwa-service-worker**

**cd project-name**

**like**

**cd angular-pwa-service-worker**

**ng add @angular/material**

**button, textfield, passwordfield, card, table etc.**

**According angular/material for every html component they created modules. So we have to import the modules depending upon what type of angular/material component using the application.**

**Card module : <mat-card> it is like a container which is use to insert text, images, contents, actions etc.**

**<mat-card> contains**

**<mat-card-title>**

**<mat-card-subtitle>**

**<mat-card-content>**

**<mat-card-header>**

**<mat-card-action>**

**Node js provided pre-defined module json-server which help to run static json file as a server. So using Angular we can add, delete, update and retrieve json data from static json file.**

**To install json-server in local machine we have to run the command as**

**npm install –g json-server**

**to run json file as a server we have use the command as**

**json-server filename.json**

**Now create the component**

**ng g c employee**

**Create service**

**ng g s employee**

**According to Angular material**

MatFormFieldModule

**Wrap label and textfield, passwordfield, radiobutton, checkbox etc.**

  //post method for insert

  //post method takes 2 parameter

  //1st parameter url and 2nd parameter json data.

**Angular running on port number 4200 (with in build angular web server).**

**Json-server running on port 3000.**

**Angular development mode project running on port number 4200.**

**Production mode or environment (means actual server, live server).**

**In production environment server are**

**Like tomcat, web logic, IIS, Jboss etc.**

**After development we have to build the projects.**

**Command to build the project**

**Before build please stop the project**

**And run the command as**

**ng build –prod**

**after build**

**move insider a folder**

**cd dist**

**cd product-folder-name**

**now with the help of node js we will create small server and deploy this application in that server.**

**npm install –g http-server**

**now we will run angular project in http-server**

**http-server –o (this command must be execute where index.html file present after build the project).**

**Angular/material**

**Pwa features**

**Bootstrap is use to create first CSS framework for mobile application. Bootstrap is use responsive base upon device like Laptop, Desktop, Mobile etc.**

**Angular PWA provide responsive as well as progressive features.**

**If application providing progressive features it work on offline, it support native application, it support push message features.**

**Now we are going add service worker features.**

**Delete dist folder**

**We have to add service worker features in project**

**ng add @angular/pwa**

**Phase 3 :**

**Day 17 : 01-04-2021**

**Html**

**CSS**

**JavaScript**

**Bootstrap**

**jQuery : Library**

**TypeScript**

**Angular : Framework**

**React JS**

**Node JS : Node JS not library nor a framework. It is a run time environment for JavaScript program or library or framework.**

**HTML/CSS/JavaScript --------🡪 Java**

**jQuery JEE**

**Servlet/JSP/EJB**

**Asp.net**

**Php**

**Python**

**Node JS**

**Node JS : contains lot of modules. With help of those modules we can create client side as well as server side programming language.**

**After node JS JavaScript can use server side scripting language.**

**CSS SSS**

**Core modules**

**External or Third party modules.**

**“Node JS’ goal is to provide an easy way to build scalable networking program”**

**JavaScript is open source.**

**Node JS is open source, cross platform runtime environment.**

**It provide an event driven architecture and non blocking IO operation.**

**JavaScript is a single thread.**

**Java**

**.net**

**Server : Tomcat, Web Logic, JBoss**

**IIS :.net**

**Multithreading**

**By default all java or .net server are multi threaded.**

**Program : set of instruction**

**Process : time taken to execute the code.**

**Processor : responsible to execute code.**

**Thread : small execution of a code within a process.**

**Using Java or .net we can create more than on create to do multi tasking.**

**Tomcat Server : Thread base.**

**100**

**101**

**Java -🡪 JRE**

**JavaScript 🡪 Node JS**

**Node JS contains lot of modules. Some module is part of Node JS software some modules we have to download using NPM.**

**Using http or express module we can create Server side scripting language.**

**MEAN : Mongo DB Express Angular Node JS**

**MERN : Mongo DB Express React JS Node JS**

**REPL : Read Eval Print Loop**

**We can’t use document as well as window object in REPL (terminal).**

**Node JS provided two pre-defined global objects.**

**console.**

**process.**

**In JavaScript using ES5**

**window.document.write(“Welcome to JS Page”);**

**document.write(“Welcome to JS Page”);**

**it is use to display the output on web page.**

**\_\_filename**

**\_\_dirname**

**Pre-defined global property.**

**Node JS Core Module**

**fs module (file system).**

**Using fs module we can do file handling program like read, write and append files.**

**Syntax to load the modules**

**var referenceName = require(“moduleName”);**

**Or**

**let referenceName = require(“moduleName”);**

**to install external module we have to use the command as**

**npm install –g moduleName**

**tsc**

**@angular/cli**

**setTimeout()**

**setInterval()**

**clearInterval()**

**pre-defined asynchronous operation.**

**Promise then().catch()**

**Observable .subscribe(next,error,())**

**Day 18 : 02-04-2021**

**Reading data from console using node js**

**http**

**express**

**readline : It is a external node js module which help to take the value through console.**

**Ask user how many records do you want to store.**

**After confirmation take id,name,salary and store in file as json data.**

**You have to read any one record values**

**Or display number of records stored in file.**

**Debug the JS (JavaScript, Angular, Node JS).**

**JavaScript ES5 or ES6 provided keyword as debugger to debug the application.**

**Create simple file with break point using debugger keyword.**

console.log("1st statement");

console.log("2nd statement");

debugger;

console.log("3rd statement");

console.log("4th statement");

debugger;

console.log("5th statement");

console.log("6th statement");

debugger;

console.log("7th statement");

console.log("8th statement");

debugger;

console.log("9th statement");

To run the application in debugger mode use command as

Node debug/inspect applicationName.js

To check the code line by line we have to use the command as

Next :

If we want to move one break to another break we have to use the keywords as

Cont

To come out from debugger terminal we have to use the command ad Cntr + C twice

**Node JS exports modules.**

**Typescript import and export**

**The module.exports is a special object which include in every JavaScript file in the Node js application by default.**

**The module is a variable that represents the current module and export is an object that will be exposed as a module.**

**Module : collection of function, variable, classes, interfaces, enum etc.**

**Model ; it is a TS class**

**Day 19 : 05-04-2021**

**Asynchronous in Node**

**Util module**

**http module**

**synchronous and asynchronous communication.**

**ES5 and ES6**

**setInterval()**

**setTimeout()**

**clearInterval()**

**Promise()**

**Observable() :rxJS**

**HttpClient**

**Using Plain JavaScript load the data from fake api and display in table format.**

**Using promise and document.getElementById()**

**Node JS util module**

**Node JS contains core util module which help to do utility task. Like data formatting, validation etc.**

**http module : http is a core module part of node js which help to create the server.**

**Using http module we can start the server on specific port number we can receive the request from a client and we can give the response back to the client.**

**Servlet/JSP**

**Asp.net**

**Php**

**After node using JavaScript we can create the own server.**

**http module**

**URL /favicon.ico**

**URL /**

**Etc**

**Day 20 : 06-04-2021**

**URL : Uniform resource locator**

[**http://localhost:9090/aboutus**](http://localhost:9090/aboutus)

**if methods is GET**

**if want to pass the data through URL we have to use query param**

**http://localhost:9090?name=Ravi**

[**http://localhost:9090?name=Ravi&age=21**](http://localhost:9090?name=Ravi&age=21)

**we can take the form tag with get methods.**

**url : it is core module which help to get the details about the URL.**

**Using http module we create the server.**

**http module provide basic server features.**

**http : Hyper text transfer protocol.**

**http or https**

**http protocol methods**

**according http it must be support http methods.**

**Like**

**Entity or table or document**

**Employee or Person or Login or Account**

**Entity is known as resources.**

**get() : get the resources**

**: get all employee details,**

**: get Employee using Id**

**: get Employee using designation**

**Select all records**

**Select records depending specific property.**

**post() : create the resources**

**Store the data.**

**Insert the records.**

**put() : update the resources**

**Update the salary using empid**

**Update the age using empid**

**Update all details.**

**delete() : delete the resources**

**: delete the records using property as id**

**CRUD Operation (Create, Read, Update and Delete)**

**Create –Post()**

**Read –Get()**

**Update –put()**

**Delete – delete()**

**Before Node JS**

**Java (Servlet , JSP)**

**Asp.net**

**Php**

**Python**

**View : HTML/CSS/JavaScript ES5**

**Forms**

**View must be Ajax with jQuery or without jQuery**

**We can call get(), post(), put() and delete() methods.**

**XMLHttpRequest**

**Java, .net , php, python, node js**

**Req(http/https) req**

**Client HDFC HSBC**

**Res(http/https) Node asp.net**

**Java**

**Web Service :**

**Giving the Service for web application when both application running using different technologies, different OS.**

**2 types of Web Service**

1. **SOAP Web Service**

**Simple Object Access Protocol.**

**SOA ( Service Oriented Architecture).**

**SB UDDI Registry**

**Wsdl**

**Web service description**

**Language : XML**

**Lookup Register the service**

**SR SOAP Req SP**

**SC SOAP Res methods**

**Service Requester asp.net**

**Service Consumer node js**

**Java**

**SOAP Web service consumer and produce the data in the form of only XML.**

**SOAP web service is standard web service.**

**SOAP internally follow its own structure.**

1. **RESTFull Web Service**

**REST full web service we can consume as well as produce the data in the form of XML as well as non xml like JSON, text, html, buffer, media type etc.**

**Restfull web service expose resources or technologies like Servlet, JSP, Node JS(Express Module), asp.net, php, python as a Web Service using URL.**

**JSON or XML(DTD or XSD).**

**Document Type definition**

**Xml Schema definition.**

**Express :**

**WADL : Web Application Description language**

**Swagger API: to create documentation for REST API.**

**MEAN : Mongo DB Express Angular Node**

**ng new component-communication**

**with routing**

**Sharing the data between two component**

1. **Parent -> Child : @Input decorator help to pass the value from parent component to child component.**
2. **Child -🡪 Parent : @Output decorator help to pass the value from child to parent component.**

**@Output with EventEmitter API**

**@ViewChild()   
Using ViewChild decorator we can create the reference using inject to access child component property and functions.**

1. **Sibling component**

**HTML5/JavaScript**

**sessionStorage**

**localStorage**

**Using Angular**

**Create SharedService class**

**One component set the value**

**Another get the value.**

**Shared data using routing**

**Create new project with routing features**

**ng new angular-routing-shared-info**

**ng g c login**

**ng g c success**

**ng g c dashboard**

**ng g c info**

**08-04-2021**

**Create the server we use http module**

**http is a core module which provide basic functionality.**

**There are various third party modules provide by different vendor. Which help to develop web application as well as REST API using Node JS(JavaScript).**

**Express**

**Koa**

**Hapi.js**

**Geddy**

**Etc**

**Express js (module) help to create the web application as well as REST API. It support all Http method. Get,post, put and delete. It consumer as well as produce JSON data.**

**Npm install –g moduleName (globally )**

**Or**

**npm install moduleName (locally)**

**package.json file**

**This file provide all modules or dependencies details for that projects.**

**create new projects (new folder)**

**one project in VS Code.**

**To create the package.json file using command as**

**npm init or npm init –y**

**To install the express module**

**npm install express**

**To restart if we do any change in node js file we have to install one external**

**nodemon**

**npm install –g nodemon**

**After install successfully**

**nodemon filename.js**

**rather than node file.js**

**GeT Method : Get method body data is empty**

1. **Get method we can call through URL**
2. **Get method we can call through hyperlink**
3. **Get method we can call through HTML/HTML5 forms with Get method.**
4. **We can call from angular using HttpClient get methods.**

http://localhost:9090

http://localhost:9090/aboutus

http://localhost:9090/contactus

http://localhost:9090/login

http://localhost:9090/singleQuery?name=Ajay

<http://localhost:9090/multiple?id=100&name=Ramesh&salary=25000>

// http://localhost:9090/singlePath/Mahesh

// http://localhost:9090/multiplePath/100/Ravi/25000

Passing the value through URL using Get method

1. Query param concept.

URL?key1=value1&key2=value2&key3=value3

1. Path params

URL/subPath/value1/value2/value3

Query Param : If View Or presentation is HTML/HTML5 forms. When we click on submit button with get method (default HTML form methods is Get). HTML use query param concept.

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

Path Param : If view is non – GUI Application like Unix or command prompt. They can use path param concept to pass the value to server (node js or any web application).

Angular and React JS use more path param concept to pass the value.

CURL :

In Angular forms template driven form or model driven form template pass the value to component and component pass the value to user – defined service and user- defined service take the help of HttpClient to Get method to pas the value to Node JS or Rest API

Consider from Angular side

this.httpClient.get(“url?id=”+idvalue+”&name=”+nameValue)

: query param

this.httpClient.get(“url/”+idValue+”/”+nameValue)

: path param

**Post Method: Data will send through body part of http request.**

1. **Post method we can call through HTML/HTML5 forms with Post method.**
2. **We can call from angular using HttpClient post methods.**
3. **Browser provide some plugin to test post, put and delete methods.**

**Like postman client or any other plugin.**

**In Node js to get the data from request body we have to add external module as**

**body-parser**

**09-04-2020**

**Create folder**

**Express-login**

**Create package.json file using**

**npm init**

**install two modules**

**npm install express body-parser**

**CRUD Operation : without DB (Using Array)**

**REST API**

**Get**

**Post**

**Put**

**Delete**

**Entity or Resource**

**Customer, Employee, Product etc.**

**Customer**

**custId, cName, age**

**Create new folder or directory with name**

**Customer CRUD Operation – Express**

**Create package.json file**

**Install two modules**

**Npm install express body-parser**

**12-04-2021**

**Socket IO programming (chatting application).**

**Using socket we can share the data through net environment using API(Java, Python, .net, Node).**

**In node JS net is a type of core module which help to achieve TCP(Transfer Control Protocol) and UDP(Use Data Protocol) interface.**

**It allow use to make TCP or UDP connection to some endpoint and using that connection we can send and receive the data from one end point to another end point.**

**Using TCP : Client and Server application.**

**TCP : Using command prompt. No UI.**

**WebSocket : Socket programming using Web application.**

**WebSocket is the technologies which help to send the data from UI.**

**(HTML,JavaScript,Angular etc).**

**WebSocket internally use their own protocol with the help of Http protocol. WS (Web Socket).**

**Socket.io**

**Websocket is a technologies.**

**where Socket.io is a library base upon the websocket.**

**Socket.io do the abstraction for to send the data from web application to client.**

**13-04-2021**

**Mongo DB (Database )**

**MySQL or Oracle or No SQL (Mongo DB)**

**Mongo DB is a document – oriented No SQL Database or No Relational Database use for high volume data storage.**

**In Database : Data, Information, Database, DBMS, RDBMS**

**Relational Database management System.**

**File base system**

**Database system (Tables )**

**According to RDBMS**

**Table (Relation) -🡪 Schema**

**Column (Attribute)**

**Employee -🡪 id , name, salary -🡪 database to store the value in attribute.**

**Employee**

**Id(PK) Name Salary**

**100 Raj 12000**

**101 Meeta 14000 Age**

**102 Veeta Tester**

**In Mongo DB we use collection like a tables in SQL databases. Collection hold more than one documents like a records in SQL database. And Every document independent with unique identity.**

**Mongo DB help store the information in form key-value pairs like json.**

**MEAN**

**MERN**

**Node JS (Express) module -🡪 MySQL (Relational Database).**

**Where value may be number, string, Boolean, data, complex data or array.**

**HBase, Cassandra, MongoDb, redis, GraphDB,**

**To Start the mondodb service we have to run the command as**

**Mongod (default path to create server related files is C:\\data\db)**

**Inside a C drive**

**Inside bin folder**

**mongod (default path C:\\data\db)**

**mongod –-dbpath D:\\mydb**

**mongod : To run the service**

**mongo : to run the mongo database terminal**

**Control + L command to clear old output from terminal**

**Mongo DB**

**Database : database are groups of collections(tables).**

**show dbs; This command is use to display all database**

**show databases : this command is use to display all databases.**

**Create user-defined databases**

**use databasesName : this command create new database if database is not present and move inside that database(point the that database).**

**use meanstack**

**Syntax to create the collection (tables)**

**db.createCollection(“collectionName”)**

**To display all collection or tables present in database**

**show collections**

**or**

**show tables**

**To insert documents or create document inside a collection**

**db.collectionName.insert({key1:value1,key2:value2,key2:value3});**

**key must be string type and value may be number, boolean, string, data, array or complex data.**

**db.sample.insert({name:"Ravi Kumar",age:21});**

**View all documents(records) from collections**

**db.collectionName.find();**

**By default for every document json created default property as \_id value unique values.**

**If you want to pass user-defined value for \_id (pre-defined attribute) when we can insert it.**

**We will insert 4 documents in Emp collection with \_id user-defined values.**

**{ "\_id" : 2, "name" : "Mahesh", "salary" : 16000 }**

**Insert Array types of values.**

**db.Emp.insert({\_id:4,name:"Balaji",salary:34000,skillSet:["HTML5","CSS3","Angular","NodeJS"]});**

**Insert Complex values with one property**

**db.Emp.insert({\_id:5,name:"Seeta",salary:32000,skillSet:["HTML5"],address:{city:"Bangalore"}});**

**Insert Complex values with more property**

**db.Emp.insert({\_id:6,name:"Veeta",salary:30000,skillSet:["HTML5","CSS3"],address:{city:"Mumbai",state:"Mh"}});**

**Insert complex with Array property**

**db.Emp.insert({\_id:7,name:"Keeta",salary:35000,skillSet:["Python","Java"],address:[{city:"Mumbai",state:"Mh"},{city:"Bangalore",state:"Kar"}]});**

**To insert many document in one query.**

**We can insert the document (records) in collection if collection doesn’t present it create and insert the document in collection.**

**db.Employee.insert({\_id:100,name:"Ravi",age:21});**

**db.Employee.insertMany([{\_id:103,name:"Mahesh",age:25},{\_id:104,name:"Raju",age:32}]);**

**retrieve specific records using index position**

**db.Employee.find()[0]; 0 is index position**

**retrieve specific position specific attribute value.**

**db.Employee.find()[3].\_id;**

**db.Employee.find()[0].name;**

**Retrieve more than properties (Attributes) values.**

**db.Employee.find({},{propertyName:+ve/-venumber});**

**In find method 1st curly braces for conditions. 2nd curly braces to retrieve properties. PropertyName : +ve/-ve that property display By default**

**+ve /- ve : true and 0 false.**

**By default \_id property is true.**

**db.Employee.find({},{name:1});**

**name and \_id**

**By default pre-defined property \_id 1 consider.**

**db.Employee.find({},{age:1});**

**find() with first curly braces with conditions.**

**\_id is equal to 100**

**db.Employee.find({\_id:100});**

**name is equal to Lokesh**

**db.Employee.find({name:"Lokesh"});**

**Age is equal to 25**

**db.Employee.find({age:25});**

**Relational operator**

**db.Employee.find({age:{$gt:24}});**

**$gt >**

**$gte >=**

**$lt <**

**$lte <=**

**$eq =**

**$ne !=**

**Update document property using without using any property.**

**db.CollectionName.update({condition},{$set:{key:value}});**

**update one property value using one condition**

**db.Employee.update({\_id:100},{$set:{age:30}});**

**update one property value using both conditions must be satisfies.**

**db.Employee.update({$and:[{\_id:101},{name:"Ramesh"}]},{$set:{age:35}});**

**any one records updated but only one records where the condition satisfies**

**db.Employee.update({$or:[{\_id:101},{name:"Raju"}]},{$set:{age:65}});**

**Multiple records update if conditions satisfies.**

**db.Employee.update({$or:[{\_id:101},{name:"Raju"}]},{$set:{age:75}},{multi:true});**

**14-04-2021**

**Remove/delete the records from collection**

**This command remove all records(documents) from collections.**

**db.collectionName.remove({})**

**db.collectionName.remove({\_id:1})**

**db.collectionName.remove({salary:{$gt:12000}})**

**create and insert**

**db.createCollection(“Employee”)**

**db.Employee.insert({})**

**Insert directly**

**db.Employee.insert({})**

**Mongo Db Collection Relationship**

**One to Many**

**One to One**

**Relationship**

**Using 2 ways**

1. **Embedded Collection**
2. **Linking Collection**

**Using Embedded style one to one**

**Employee – Address**

**Employee – Project**

**Employee**

**\_id 100 Name Ravi age 21**

**Address complex**

**City state pincode**

**Bangalore Kar 560096**

**Array of complex**

**Projects Pid ProjectName**

**PId ProjectName**

**Using Linking collection**

**One to many or one to one**

**Trainer**

**Student**

**Connecting mongo DB database using node js using mongodb module**

**npm install mongodb**

[**http://localhost:9090/displayProduct**](http://localhost:9090/displayProduct)

**combine express js + mongodb program**

**mongoose module to do CRUD Operation with MongoDB**

**ORM (Object Relation Mapping ) In Node JS**

**15-04-2021**

**Mongoose : Mongoose is a open source Object Data**

**Modelling (ORM like Hibernate or JPA) library for Mongo DB and Node JS.**

**Mongodb mongoose**

**mongodb : native driver in node JS to interact with MongoDB(Database). Using mongodb we can store the document or records without schema.**

**Mongoose : it provide Schema as well as Model which help to map json records with documents.**

**Schema : By default Mongo DB (Database ) is schema less. Using Mongoose we can create schema which help to create the structure for the document.**

**Number of attribute as well as type of attribute.**

**(Validate the rules).**

**Model : using schema we create the reference or instances for the document to store the records in databases.**

**MEAN Stack Project**

**Frontend -🡪 Angular**

**Backend -🡪 node , express and mongoose and mongodb**

**MEAN Stack**

**MVC : Model View Controller**

**View --🡪 Angular Project**

**Controller -🡪**

**Model 🡪**

**View -🡪 Angular(template—>component—>user-defined service 🡪 http service)**

**Path : Get/Post/Put/Delete**

**Express, bodyParser, mongoose etc**

**App.js/Server.js -🡪 Server running**

**Express Routing -🡪 Base upon the path redirect to specific controller.**

**Controller 🡪 Take the help Model**

**Model -🡪 Created using Schema**

**With the help of Model we interact with MongoDB database to do CRUD Operation.**

**Please backend folder**

**Then create three folder**

**Controller**

**Model**

**Router**

**app.js**

**Then create the package.json file**

**Using npm init**

**npm install express body-parser mongoose**

**view -🡪 app.js 🡪 router -🡪 controller -🡪 modell🡪 dabase (mongodb)**

**Create angular project in frontend folder**

**ng new product-crud-operation**

**routing : yes**

**16-04-2021**

**View (Angular )**

**Frontend**

**Template -🡪Component -🡪 User – Defined Service**

**-🡪 HttpClient (Get/Post/Put/Delete)--🡪**

**REST API call to consume and produce the data.**

**Backend**

**App.js or server.js (node js)--🡪Router ---🡪Controller --🡪 Model --🡪 Database (MongoDB)**

**Check the URL post, put and delete by postman client**

**Write URL**

**Select method : post/put/delete**

**In header option**

**Key 🡪 content-type**

**Value 🡪 application/json**

**Body part : raw and write proper json data**

**Click on button.**

**Frontend**

**Create Angular project**

**ng new angular-crud-operation**

**create four component and one service**

**ng g c store-product**

**ng g c retrieve-product**

**ng g c delete-product-by-id**

**ng g c update-product-price**

**ng g c retrieve-by-id**

**ng g s product**

**CORS Origin :**

**Cross Origin Resource Sharing**

**Two domain communicate to each other**

**Angular Node JS(Express Module**

**4200 9090**

**To avoid this error we have to install**

**cors module in node JS**

**app.user(cors())**

**20-04-2021**

**HttpClient get(), post(), put() and delete method return type is Observable and format of data in json format.**

**If node js (express module) return data or message in json format you no need to do any change in angular**

**If return type is json from express module**

**http.get(“url”** {responseType:'json'}**)**

**http.post(“url”,jsonData)**

**http.detele(“url”)**

**http.put(“url”,jsonData)**

**if return type is non-json format**

**http.get(“url”,** {responseType:'text'}**)**

**http.post(“url”,jsonData,** {responseType:'text'}**)**

**http.detele(“url”,** {responseType:'text'}**)**

**http.put(“url”,jsonData,** {responseType:'text'}**)**

**Phase 4 :**

**Docker : Docker is a advanced OS Virtualization software platform that makes it easier to create, deploy and run the application in a Docker Container.**

**Virtualization : Virtualization is an abstract version of physical machine.**

**Window OS**

**Mac OS**

**Unix or Linux OS**

**Docker Container : Docker container is very**

**Light weighted package that allow the developer to package up an application and deploy it as one with the help of in build libraries and other dependencies.**

**MEAN Stack : Angular, Node JS and Mongo DB**

**Node JS**

**Using Npm**

**Ng**

**Mongo db**

**Npm express**

**Docker image : The file system and configuration of our application which is uses to create the container.**

**Docker image are the source code for our container.**

**Docker registries : Docker stores the images, and build the image. Publish those image in Docker hub registries.**

**public(more than one) and private (one).**

**Docker software.**

**2 options**

1. **Install the Docker in local machine ie window, mac or Unix etc.**

**Open the command or terminal the check the**

**Docker version**

**wsl\_update\_x64 (window os install kernel using this software).**

**docker –version**

**docker images**

**docker login**

**dockerId**

**password**

1. **You can use lms virtual lab. (Please launch the lab before start the session).**

**Docker hub account login**

**Using admin command prompt**

**C:\Program Files\Docker\Docker\DockerCli.exe -SwitchDaemon**

**21-04-2021**

**Docker : local machine**

**docker command**

**Docker : Virtual Lab machine or Unix or**

**Start all command using**

**sudo Docker command**

**docker image : It contains file configuration of our application which are used to create or run the container.**

**Commands**

1. **docker –version**
2. **docker images**
3. **docker pull image-name (hello-world)**
4. **docker login** 
   1. **provider dockerId and password**
5. **docker run imageName/imageId**

**Docker images**

* + - 1. **hello-world (C Language)**

**2. busybox (Unix image)**

**-it iterative mode**

**Or**

**docker run busybox ls**

* + - 1. **alpine (Unix images)**

**To create user-defined image we have to create the file with name dockerFile (without extension)**

1. **Creating user-defined images base upon busybox image to execute Unix commands.**

**Docker required Unix OS (alpine) and install required software and deploy the and run the application.**

**MEAN Stack**

1. **Running simple node js(Express Module) program using Docker**

**First create npm init (package.json)**

**Install required module like**

**npm install express**

**then write app.js file**

**dockerFile**

**FROM node:alpine**

**WORKDIR /usr/src/app**

**COPY package.json /usr/src/app**

**RUN npm install**

**COPY . /usr/src/app**

**CMD ["node","app.js"]**

**Create image using command as**

**docker build -t my-express . -f dockerFile.txt**

**to check container details**

**docker container ps**

**or**

**docker ps**

**run the container using command as**

**docker run -d -p 9999:9999 my-express**

**yellow color port number where application running (it is listen in app.js file)**

**red color port number your going to host the app.**

**To delete images**

**docker rmi imageName/imageId**

**docker rmi –f imageName/imageId (forcefully)**

**To delete container**

**docker rm containerId**

**docker rm –f contaierId**

**To stop container**

**docker stop contaienrId**

**To start container**

**docker start containerId**

**To get the container details**

**docker ps**

**or**

**docker container ps**

**How to deploy angular project in Docker**

**First create angular project**

**And build the project using command as**

**ng build --prod**

**Then create dockerFile**

**FROM nginx:alpine**

**COPY /dist/angular-with-docker /usr/share/nginx/html**

**Then create the image**

**docker build -t angular-web . -f dockerFile.txt**

**nginx server by default port number is 80**

**Then we have to run the web app in container using command as**

**docker run -d -p 9898:80 angular-web**

[**http://localhost:9898/**](http://localhost:9898/)

**Jenkin**

**docker pull jenkins/jenkins**

**docker run –d –p 8080:8080 jenkins/jenkins**

**run** [**http://localhost:8080**](http://localhost:8080)

**Then it will ask password**

**Using docker ps please find the containderId for Jenkin**

**docker logs containerId**

**docker ps**

**then install the plugin**

**after that it ask profile details**

**username**

**password**

**emailId**

**To run Jenkin server (this command is use to start the Docker container)**

**docker run –d –p 8080:8080 jenkins/Jenkins**

**Open the browser and writ the url as**

[**http://locahost:8080**](http://locahost:8080)

**Jenkin :**

**Every member working in different module of project or application.**

**Every individual machine code may be work fine.**

**But when we integration phase( combine more than one person code in one machine).**

**Sample project M1 pull (master) --branch task1**

**M2 pull (master)**

**GIT M3 pull (master)**

**M4 pull (master)**

**Repo M5 pull (mater)**

**Pull in master/main branch and push user-defined branch**

**GIT**

**Write the code in their machine**

**Push the code the in git**

**Merge the code in git**

**If any changes again commit, push and merge.**

**We have to do manually.**

**Compile code, build the project, test the project and deploy the projects.**

**CI and CD tool (Continuous integration and delivery or deployment).**

**Where CI and CD toll configure with git if any changes happen the pull the code from git, build and project, test it and deploy on server.**

**Jenkin : Jenkin is a open source automation server written in Java. So with the help of Jenkin the integration phase run very smoothly.**

**Jenkin detects changes in sub version (git), then perform the task, repeatedly. (compile, build, test, deploy).**

**23-04-2021**

**1st half :**

**Project description and common git account for every team.**

**2nd half**

**Jenkin pipeline**

**AWS : Overview**

**Jenkin we are running in docker.**

**Jenkin Pipe Line**

**Jenkin pipeline is a collection of more than jobs or events.**

**Like**

**Build, deploy , test and release etc.**

**Jenkin file we can write in jenkin or in git.**

**This file contains more than jobs task details.**

**Using Groovy script**

**pipeline {**

**agent any**

**stages {**

**stage(“Check Version”){**

**steps {**

**echo “Checking Version”**

**}**

**}**

**stage(“build”){**

**steps {**

**echo “building the application”**

**}**

**}**

**stage(“Test”){**

**steps {**

**echo “Testing the application”**

**}**

**}**

**stage(“Release the version”){**

**steps {**

**echo “Release the Version”**

**}**

**}**

**}**

**}**

[**https://www.surveymonkey.com/r/NBKJBF7?trainer=akash%20kale&company=Tata%20Consultancy%20Services&milestone\_id=00060605\_01&course=phase%203%20-%20create%20scalable%20and%20dynamic%20websites&geo=US&type=ELVC&Webex\_Session\_ID=1847696173**](https://www.surveymonkey.com/r/NBKJBF7?trainer=akash%20kale&company=Tata%20Consultancy%20Services&milestone_id=00060605_01&course=phase%203%20-%20create%20scalable%20and%20dynamic%20websites&geo=US&type=ELVC&Webex_Session_ID=1847696173)

**https://www.surveymonkey.com/r/NBKJBF7?trainer=akash%20kale&company=Tata%20Consultancy%20Services&milestone\_id=00060460\_01&course=phase%202%20-%20create%20websites%20using%20front%20and%20back-end%20frameworks&geo=US&type=ELVC&Webex\_Session\_ID=1840928935**