**MEAN Stack**

**Phase 1**

**Day 1**

**26-07-2021**

<https://github.com/Kaleakash/MEAN_Stack_TCS_July_2021_Batch.git>

MEAN Stack :

Mongo Db / MySQL Express JS Angular Framework Node JS

Phase 1

Git

HTML,CSS,JavaScript using ES5

Bootstrap

Section end and phase end projects.

Phase 2

Node JS Overview

TypeScript using ES6 features

Angular Framework

Section end and phase end projects

Phase 3

Node JS

Node JS modules http, util, fs, express js

Mongo DB database : No SQL Database

Mongodb and mongoose modules to connect the database through JavaScript (Node JS)

Socket programming

Section end and phase end projects

Phase 4

Docker

AWS Overview : EC2 and S3

Deploy the MEAN Stack project in EC2

Section end and phase end projects

Capstone project : Team 5 people

GIT :

Local Version control :

SVN:

Git is sub version control system.

Version control system that records changes on files or project or application.

Merge the two team code in one application.

Git provide local as well as remote repository (folder or directory).

**Open the terminal**

Check the git version using command as

Open the Terminal

git --version

Then create folder

Move inside a folder.

mkdir folderName

Then create file using command prompt or GUI and write some contents.

To create local git repository using command as

git init

to check the status of last command we have to use command as

git status

to add the untrack file from file system to staging area.

git add filename

git status

After this command file will move from file system to staging area.

To move file from staging area to local repository we have to run the command as

git commit –m “created first file”

create github account with your

git config --global user.email "you@example.com"git config --global user.name "Your Name"

if we do any changes in existing file or added new file or folder

Then run the command as

git status

git add .

git commit –m “commit message”

These command repeat again and again.

remote repository : github, aws(code commit), azure etc.

To push the data from local repository to remote repository we have to use the command as

git remote add origin URL

git push –u origin master/man

or

git push

or

git push –u origin HEAD

Another way to create the repository

git clone URL

if first time we want to download the data from remote repository we have to use

git clone URL

**Phase 1**

**Day 2**

**27-07-2021**

**git branch : git branch is a like pointer which hold more than one commit details.**

**Git commit : it is use to send the data from staging area to local repository is known a git commit.**

**Git staging are : it a buffer area created by git which hold the data before commit. It is a intermediate layer or memory between local system and local repository.**

**By default depending upon the version of git default branch created it may main or master**

Default master/main branch

Do some changes 🡪add/commit -🡪add/commit --🡪add/commit

A branch

Do some changes 🡪add/commit --🡪add/commit

After done all changes in user-defined if code is correct then merge user-defined branch into main/master branch else we will delete the branch.

To check default as well as user-defined branch names

git branch

To create new branch

git branch branchName

To switch from one branch to another branch

git checkout branchName

To merge user-defined branch code to current branch ie main/master

git merge user-definedBranchName

To delete user-defined branch

git branch –D branchName

git pull : it use to download latest data from existing remote repository.

git pull execute in main/master branch if you want to do changes even space or dot. Please create user-defined branch do the changes if any thing wrong switch to main/master branch and delete user-defined branch.

UI Technologies

Day 2 and Day 3 HTML/CSS

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

http : protocol : hyper text transfer protocol : secure

www : world wide web

google : domain

com : commercial

req(http/https)------🡪

Client Server

🡨---res(http/https)----- HTML/HTML5

CSS/CSS3

JS (JavaScript)

**HTML/HTML5-**🡪 It use to display the content on browser.

CSS/CSS -🡪 Apply good look and feel or presentation logic on contents.

JavaScript 🡪 Event on contents or programming on web page.

basically if a web page was a body then html is the skeleton, css is the skin, and javascript is the organs

HTML : Hyper text Mark up language : it is use to create web page it may be static or dynamic.

HTML provide lot pre-defined tags or elements. HTML is not a case sensitive as well as not a structure.

Tag syntax

<tagName> opening tag

</tagName> closing tag

<tagName/> self closing tag

1. Html
2. Head
3. Body
4. Title
5. P

Open the notepad or any editor

Write the html code

<html>

<head>

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

</head>

<body>

<p>Welcome to My Simple Web Application </p>

</body>

</html>

Save the file with any name with extension .html

Make sure file extension must be .html

Then open in an browser.

**IDE**

Notepad ++

Bracket

ATOM

Eclipse

VS

**VS code**

Break tag <br/>

Heading tags

H1 to h6 heading tags

H1 means largest

H6 means smallest

Html 4 version (xhtml)

<!doctype html public url=”pathpath.**dtd**”>

Document type definition

dtd file contains the rules what is root tag name ie html, which contains two child tag head and body

body tag can contains more than one p as well as other tags.

html5 they remove dtd file

**<!doctype html> : this tag is use to give the instruction to browser we are going to html5 features this tag also optional.**

And added more tag to make html dynamic web page without depend on any other language.

**Hyperlink** : hyperlink is use to connect more than one web page.

<a href=”pageName/pageName.html”>Text</a>

a : anchor tag

href : hyper reference.

To add the image

Syntax

<img src=”ImageName.jpeg/gif/” />

Img : image

Src : source

**Phase 1**

**Day 3**

**28-07-2021**

List Tags

HTML provide different type of list tags

UnOrder list : ul : unorder list and li :list item

Order List : ol order list and li list item

Definition List dl : definition list, dt : definition term and dd : definition description

**Table Tags**

Table tag,

Tr : table row

Th : table heading

Td : table data

**EmpId Name Age heading**

100 Raj 21

101 Seeta 22

102 Meeta 23

**Attribute** : attribute is use to describe the properties of a tags.

Attribute we can use in the form of key-value pairs.

We can use value may in single quote or double quote or without quote.

We have to use attribute inside a opening tags.

Syntax

<tagname name1=”value1” name2=’value2’ name3=value3> </tagName>

**Form tags**

Login Page

UserName TextField

Password PasswordField

Submit Reset

Before HTML5

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

After HTML5

<input type=”number/email/date/url” />

HTML form by default method is consider as GET

If method is GET data send through URL using URL re-direct technique

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

Get method is not a secure.

If we want data must be secure we have to must method=”post”

If method is post data send through Request body part.

Div tag

Span

CSS :

CSS provide set of properties which help to apply good look and feel for the web page.

Using HTML we have to depending upon other tags. some times we can’t achieve that good look and feel base upon the client requirement.

Using HTML actual contents and formatting style combine in one place or page.

CSS mainly divided into three types

1. Inline CSS
2. Internal CSS or embedded CSS
3. External CSS

Inline CSS

Syntax

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

Using Inline Css applying style attribute for more than one tag is more complex. In line CSS is good if we want CSS for few or only one tag.

**Internal CSS or Embedded CSS**

<style type=”text/css”>

selector {property : value;property:value;}

</style>

We have to write the style tag in between head tag.

Div is known as division tag. Div tag is also known as container tag. Which can contains more than one other tags as well as another div tag. Div tag is use to represent particular part of web page.

**Types of selectors**

1. Universal selector : \* : all tags : \*{property:value}
2. Specific tag selector : tagName {property:value}
3. Local class selector : tagname.className {property:value}
4. Global class selector .className{property:value}
5. Id selector : #idName{property:value}
6. Child selector : parent Name childTagName {property:value}

Class selector and Id selector

class : means group of tags may be same type or different type. More than one tags can contains same class name but id must be unique

<div>

<p class=”p1” id=”a1”>First</p>

<p class=”p2” id=”a2”>Second</p>

<p class=”p1” id=”a3”>Third</p>

<p class=”p2” id=”a4”>Fourth</p>

</div>

Limitation of Internal CSS. The CSS rules apply for local that web page. But if we want CSS rules for globally accessible then we have to use external CSS.

styles.css

write CSS rules

in html web page

inside head tag

<link rel=”stylesheet” type=”text/css” href=”styles.css”/>

Box Model

**Phase 1**

**Day 4**

**26-07-2021**

**Box Model**

In html Every tags ie DOM (Document Object Model). Every tag is known as DOM elements.

Internally follow box model.

Margin-left

Margin-right

Margin-top

Margin-bottom

Border-left

Border-right

Border-top

Border-bottom

Padding-left

Padding-right

Padding-top

Padding-bottom

**JavaScript :**

**JavaScript using ES5 not ES6**

**ECMA Script : European Computer Manufacture Association**

ECMA is a concept.

OOPs

One of the implementation of ES5 or ES6 is JavaScript.

JavaScript was object based interpreter scripting language.

Object based Vs Object Oriented

JavaScript contains lot pre-defined object as well as we can create user-defined object not class concept.

Interpreter Vs Compiler :

Interpreter : check the code line by line

Compiler : it check all the code at time and convert into another format.

Using JavaScript we can do programming on web page.

If we want to write JavaScript code we have to use the script tag.

Syntax

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

</script> closing tag

We can write more than one script tag in html in between head as well as body tags.

To display the message through JavaScript we can to use document.write(“msg”);

document is pre-defined object and write is pre-defined functions.

In JavaScript it not mandatory every statement must be end with semicolon.

Variable and datatype

In JavaScript we can declare the variable using **var** keywords.

Syntax

var variablename;

var abc; // default value of variable is **undefined**

var m=10; //it is consider as number type

var n=10.10; // number type

var name=”Ravi Kumar”; // string consider

var result = true; // Boolean type consider

var obj = new Date(); // obj is consider as object reference.

**Operators**

Arithmetic operator : +, -, \*, /, %(remainder)

Logical : &&, ||, !

Conditional operator : >, >=, <, <=, ==, ===, !=

Increment and decrement : ++, --

Type of operator : typeOf

Ternary operator : condition ? true :false;

== and ===

typeof

ternary operator :

if statement

simple if

if else

if else if

if(condition){

true

}else {

false

}

switch statement

looping

while loop

do while loop

for loop

**function**

function is use to write the set of instruction to perform a specific task.

2 types

1. Pre-defined function
2. alert() : this function is use to display the pop message.
3. prompt() : This function is use to take the value through keyword.
4. eval() : it is use to convert string to number
   1. parseInt(): it take only number without decimal
   2. parseFloat(): with decimal
5. Confirm(): it contains 2 button ok and cancel. If user click ok it return true else return false.
6. User-defined function

In JavaScript we can write function in lot of ways.

1. Normal function declaration syntax

function functionName(parameterList) {

}

**Event :** Event is interaction between user and component (dom elements) or event provide bridge between html and JavaScript code.

JavaScript lot different type of events. All event in JavaScript start with on followed by event name.

Example

onClick

onDblClick : button group : button, radio button, checkbox

onMouseOver

onMouseOut : image or other tags

onKeyUp

onKeyDown : textfield

OnSubmit : validation

onChange : dropdown

onFocus : while entering in text field

onBlur : while exit from text field

etc

**DOM : Document Object Model**

**index.html**

**DOM Hierarchy will create**

<html>

<head>

<title>Simple Web Page</title>

</head>

<body>

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

</body>

</html>

Html 🡪 root tag

Head body

Title p

textNode 🡪 Simple Web page text Node : Welcome to My Web Page

DOM API Document Object Model Application programming interface.

All language like Java, Python, C# as well as JavaScript provide DOM API which help to read, write and update dom (html tag contents ) dynamically.

**Phase 1**

**Day 15**

**30-07-2021**

**External JavaScript**

**If we want JavaScript code particularly for only one page then we can use internal JavaScript.**

**But if we want JavaScript function code can access more than one web page that time we can use external JS file**

**DOM Operation**

**Expression style function**

**Normal function we can call before declaration as well as after declaration.**

**But expression style function must be declare first then invocation.**

**Callback function : passing the function itself or function body or function name to another function as a parameter is know as callback function.**

**Arrow function : arrow function is part of ES6 features.**

**Arrow function is a short cut syntax for expression style function.**

**array :**

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

**syntax**

**var num1 = [10,20,30,40,50,60]; literal style**

**var num2 = new Array(10,20,30,40,50,60); object creation style**

**array start from index position 0 to size-1**

**array forEach() function takes callback function as a parameter which help to display the value one by one without taking help of any loop.**

**IIFE : Immediate Invoke function Expression**

**Syntax of IIFE function**

**(function declaration)(functionCall)**

**IIFE function we can’t do re-usability.**

**Array methods**

**splice(indexpostion, deleteCount, add elements/replace elements)**

**Day 6**

**02-08-2021**

**JavaScript provide two pre-defined collection class**

**It Set and Map (ES6).**

**Set : A set is a type of collection or data structure which help to store more than one value.**

**Set doesn’t allow duplicate.**

**for in loop : it retrieve the index position**

**for of loop : it retrieve value**

**Map : Map is a type of collection or data structure which help to store data in the form of key-value pairs. Key is unique and value may be duplicate.**

**ES6 features**

**let and const**

**from ES6 JavaScript we can use var,let and const keyword to declare the variables.**

**Using var keyword we can declare same variable once again ie re-declaration. Using let keyword we can’t do re-declaration.**

**var n=10;**

**n=20; re-assign**

**var n=30; re-declare**

**let m=40;**

**m=50; re-assign**

**let m=60; // Error**

**int a=10;**

**a=15;**

**int a=20; // Error**

**using var we can do global scope. But using let we can do local or block scope.**

**If we declare the variable using const we can’t change the value of that variable.**

**OOP : object oriented programming**

**objects**

**object is a any real world entity.**

**State or properties -🡪 have -🡪 variable/ fields**

**Person**

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

**Bank**

**Car**

**Animal**

**Employee**

**Customer**

**Order**

**In JavaScript we can describe the object using different ways**

**1st way using function ES5 style**

**2nd way using literal style ES5 style**

**3rd way using class style ES6 style**

**this is a keyword which refer to current object.**

**Constructor : constructor is like a special function which help to create the memory.**

**It use to do initialization purpose.**

**In ES6 class must be contains only one constructor it may be empty or parameterized.**

**Literal style object creation is use if we want object with only properties not behaviour.**

**JSON : JavaScript Object Notation**

**Req(http)-🡪 Java (Spring boot) java(req)**

**Java**

**Client Amazon Web Application XML/JSON Google Pay**

**Asp.net**

**Php/python/asp.net(res) PayTM**

**🡨---Res(http) Php**

**Net Banking**

**Cash delivery**

**Python**

**Credit card**

**Web Service : Giving the service for web application when both application running using different technologies.**

**SOAP Web Service : Simple Object Access Protocol. : We can consume and produce only in the form of XML.**

**REST Full Web Service : We can consume and produce data in any format like xml, json, text, html or any other format.**

**­**

**JSON : JavaScript Object Notation : Json Store the data in the form of key-value pairs.**

**Where key must in double quote and value may be number, string, Boolean, array, complex object. Key must be unique. Using key we can get the value.**

**{“key1”:value1,”key2”:”value2”}**

**JavaScript provide pre-defined object ie JSON.**

**JavaScript provide JSON pre-defined object which help to convert string to json and json to string or object.**

**Day 6**

**03-08-2021**

**Web Storage : HTML5 provide one of the great features ie Web Storage.**

**Which help to share the data from one js file to another js file as well data store the**

**Browser or external drive memory.**

**sessionStorage : if we store the data in sessionStorage it store till session or browser open. Once close the browser or application the value from session storage get destroy.**

**localStorage: if we store the data in localStorage it store permanently. So we have to explicitly remove data.**

**sessionStorage.setItem(“key”,value);**

**localStorage.setItem(“key”,value);**

**sessionStorage.getItem(“key”)**

**localStorage.getItem(“key”);**

**sessionStorage.removeItem(“key”);**

**localStorage.removeItem(“key”);**

**synchronous and asynchronous**

**Synchronous statement execution**

**document.write(“1st statement”)**

**document.write(“2nd statement”)**

**document.write(“3rd statement”)**

**asynchronous statement execution**

**document.write(“1st statement”)**

**--🡪 async document.write(“2nd statement”) : this code execute independently.**

**document.write(“3rd statement”)**

**2nd statement execute independently**

**synchronous function call**

**fun1();**

**fun2();**

**fun3();**

**asynchronous function call**

**fun1();**

**async fun2(); : This code execute independently**

**fun3();**

**synchronous communication**

**Client Server**

**1st Req ----🡪**

**2nd req --🡪**

**3rd req -🡪**

**Client Server**

**asynchronous communication**

**Client Server**

**1st Req ----🡪**

**2nd req --🡪**

**3rd req -🡪 both request execute independently**

**Client Server**

**JavaScript provide few pre-defined function which help to do asynchronous operation.**

**setTimeout() : it call function only once after specific period of a time.**

**setInterval() : it will call again and again base upon the time provided in 2nd parameter.**

**clearTimeout(): it is use to stop the setInterval timer**

**Ajax : Asynchronous JavaScript and XML**

**XMLHttpRequest and ActiveXObject**

**asynchronous communication**

**Client Server**

**1st Req ----🡪**

**2nd req --🡪**

**3rd req -🡪 all request execute independently**

**Client Server**

**fetch() : it is pre-defined function provide by JavaScript which internally use Ajax to send and receive the data from server.**

**fetch() function return promise objects.**

**Promise is a pre-defined object which help to handle asynchronous event of data. Promise can be resolve(success) or reject (failure).**

**Creating user-defined promise object.**

**If we want to load the data from promise object we have to use then() and catch(). If promise resolve then call else catch().**

**Day 6**

**04-08-2021**

**AJAX**

**Bootstrap**

**Without bootstrap**

**We can use inline css, internal css or external css.**

**We can user-defined class with set of property.**

**Styles.css**

**.divFontClass {**

**Font-size:**

**Font-family**

**Set property**

**}**

**Bootstrap is a open source CSS framework which provide set of pre-defined css classes with respective all dom elements. Like p, div, button, table etc.**

**Bootstrap is first open source framework which help to make web page responsive web application.**

**Now way days we can make responsive web application using bootstrap or html5 features.**

**Adding bootstrap features to web page**

1. **Through URL (CDN)**
2. **Download the bootstrap file**
3. **Using Node JS**

**Div tags bootstrap classes**

**Container and container-fluid**

**Container : it is type of bootstrap pre-defined class which provide fixed width for the web page.**

**Container-fluid : it is type of bootstrap which use full width of web page.**

**Grid layout :**

**Bootstrap provide grid layout. Which help arrange the dom element or component in row and column format. According to grid layout each row divided into 12 columns.**

**Using grid layout we can arrange html dom element or component according screen size of device.**

**Xs : extra small : <576px**

**Sm : small : >=576px**

**Md: medium : >=768px**

**Lg : large : >=992px**

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

**Form Validation**

**Using HTML5 features**

**Using JavaScript**

**When user click submit button or command button without writing username, password, min length, max length, format of phone number , email etc.**

**MEAN Stack**

**Phase 2**

**Day 1**

**06-08-2021**

**Overview of Node js**

**Typescript**

**webpack**

**babel**

**Angular Framework**

**Node JS Overview**

**jQuery**

**Ext JS**

**Coffee JS**

**Angular JS**

**Angular Framework**

**D3 JS**

**React JS**

**Vue JS**

**Etc**

**To do improvement on DOM. Read, write and update HTML content on web page.**

**Node JS is a run time environment for the JavaScript library or JavaScript framework.**

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

**Before Node JS JavaScript mainly use for client side scripting language**

**After Node JS JavaScript also known as Client side as well as Server side scripting language.**

**Before Node JS Front end technologies backend technologies**

**HTML5/CSS3/JavaScript and bootstrap ------------------🡪 Java (Spring boot)**

**Asp.net**

**Php**

**Python**

**Node**

**Front end backend**

**JavaScript**

**Node JS provide lot of pre-defined module with the help of those module we can create dynamic web application, file handling programming, REST API, connecting to database(mysql or mongo db database), security , networking using JavaScript etc.**

**REST API : Representational State Transfer Application Programming Interface.**

**Please create node js folder**

**And open the terminal**

**Check node version command**

**node --version**

**Node JS doesn’t provide document and window object.**

**TypeScript : typescript is a type of scripting language. Which also known as super set of JavaScript.**

**TypeScript all features of ES6.**

**Typescript ie ts we can’t include in html page because browser can’t understand typescript.**

**So we have to convert ts file to js file using transpiler.**

**Transpiler also type of compiler which help to convert from one file to another file format.**

**tsc ( typescript compiler).**

**With node JS by default npm command enable**

**node package manager ( using npm command we can install any external node js module).**

**Syntax to install node js external module**

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

**Syntax to install typescript external module**

**Window user**

**npm install –g typescript**

**unix as well as mac user**

**sudo npm install –g typescript**

**after install**

**now check the version of tsc**

**tsc --version**

**create separate folder with name typescript program**

**then create typescript file**

**demo.ts**

**console.log(“Welcome to Typescript”);**

**to convert ts to js fie we have to use the command as**

**tsc demo.ts**

**Typescript support data types.**

**Syntax**

**let variableName:datatype;**

**let variableName:datatype=value;**

**array : In JavaScript array can store same type as well as other types of values.**

**But in Typescript with help of data types we can create array is use to store same type of values.**

**Generic type**

**Let variableName:Array<Type>;**

**ES6 JavaScript as well as Typescript we can use in loop and of loop.**

**In loop : in loop provide the index number of array elements.**

**of loop : on of loop we can get the value .**

**Typescript function types**

1. **Function with number parameter must be match**
2. **Function with number of parameter and type of parameter must be match.**
3. **Function with specific return type like number, string, Boolean or no return type void.**
4. **Function with optional parameter**
5. **Function with default initialization**
6. **Function with rest operator or parameter.**

**MEAN Stack**

**Phase 2**

**Day 2**

**09-08-2021**

**Object and class**

**Object is a any real world entity**

**object creation using ES6 in typescript**

**parameterized constructor**

**constructor short cut initialization**

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

**class OldClassname { super class, base or parent class**

**properties**

**behaviour**

**}**

**class Newclass extends OldClassName{ sub class, derived class or child class.**

**Properties**

**Behaviour**

**}**

**With help of sub class object we can call super class all function as well as it own function.**

**class Manager extends Employee {**

**}**

**class Employee {**

**id,name,age,Address add**

**}**

**class Address {**

**}**

**class Programmer extends Employee {**

**}**

**Manager is a Employee**

**Programmer is a Employee**

**Employee has a Address**

**Typescript support interface.**

**Interface contains incomplete function or function without body.**

**Interface is use to provide specification.**

**Class always implements the interface. The class which implements interface must be provide the body for all incomplete function which belong to that interface.**

**In Typescript we use interface with properties to create the object literal is a type of interface type.**

**TypeScript support import and export concept with the help of these keyword it can allow to access one file properties and functionality in another files.**

**Modules : modules is a collection of variable, functions, classes, interfaces. Module is a like a package.**

**In typescript file name itself is consider as module.**

**export keyword is use to allow access that function, variable, class or interface in another file.**

**Typescript provide tsconfig.json configuration file which hold all configuration details for the typescript projects.**

**Syntax to create the tsconfig.json file**

**tsc --init**

**Angular Framework**

**To Create Angular Project**

**We have to enable ng command.**

**npm install –g @angular/cli**

**sudo npm install –g @angular/cli**

**After installation successfully**

**ng --version (next generation)**

**back ticks and string template :ES6 features.**

**Using back tricks we have use multi line statement and inside back ticks we have access variable or property name using string template ${variableName}**

**HTML, JavaScript and Typescript**

**Create the application using Typescript convert to JavaScript and include JS file in**

**Html page.**

**MEAN Stack**

**Phase 2**

**Day 3**

**10-08-2021**

**please check your node –version**

**ng install**

**to create new project after installation**

**ng new project-name**

**ng new demo-app**

**Routing : No**

**Styling : CSS**

**Angular JS**

**Base upon HTML, CSS, JavaScript ES5 as well as ES6 features.**

**Angular Framework 2 to 12 Version**

**HTML/HTML5,CSS/CSS3 and TypeScript and Node JS**

**UI Technologies**

**Read, Write and Update DOM properly.**

**Angular is a open source framework which help to Create SPA (Single Page Application).**

**Angular is a part of Google.**

**Library Vs Framework**

**jQuery**

**React JS**

**They are library**

**Every external library for specific purpose. Library doesn’t use for multipurpose.**

**Framework use multipurpose.**

**Library is light weighted Framework are heavy weighted.**

**Library doesn’t follow any rules and regulation. But framework allow rules and regulation.**

**Design pattern: best practise or solution for repeating problem.**

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

**If you are developing small application library is good. If you going to develop enterprise application Framework is good.**

**MVC, Singleton design pattern, component base, DI and IOC etc.**

**Multi page application**

**Index.html home.html**

**Hyperlink**

**Submit button**

**Normal button with JavaScript**

**When we move from one page to another page whole DOM get loaded in buffer memory.**

**Single Page Application**

**Rather than loaded whole page it can load only part of the page using component concept.**

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

****

**Creating angular project manually is complex.**

**Google people provided Angular CLI.**

**Command line interface.**

**Node js required**

**node --version**

**npm --version (node package manager).**

**npm install –g @angular/cli : this command is use to enable ng command.**

**sudo npm install –g @angular/cli : mac or unix users.**

**Angular**

**ng --version**

**ng new project-name**

**ng new demo-app**

**routing 🡪 no**

**styling 🡪 css**

**after project created successfully**

**move inside a project folder using cd command**

**cd project-name**

**cd demo-app**

**open the project using code . or open project in VSCode. (please open complete angular project).**

**Then to run the project use command as**

**ng serve if ask policies option yes/no : you can enter yes or no**

**after compiled project 100%**

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

**or**

**ng serve –o : After compiled 100% project it automatically open the default browser with url as** [**http://localhost:4200**](http://localhost:4200)

**open the project in VS code ( open complete angular project in VS Code with open folder option).**

**expand the src folder**

**expand the app folder**

**open app.component.html page**

**please remove pre-defined html code present in this file.**

**inside this html page please write the html code which we write**

**in body tags. Don’t write !docType, html, head, meta and body tags.**

**open app.component.ts**

**write one or more variable with default value.**

**To display those variable value in html page**

**<p>Name is {{variableName}}</p>**

**Open app.component.css : this file is like a external CSS file.**

**Decorator : it is a type of special function which help to add extra behaviour to class or properties or functions. Decorator is part of typescript. Decorator also know meta-data(data about data).**

**Decorator is like a annotation in Java.**

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

**@Component**

**@NgModel**

**@Input**

**@Output**

**@Injectable**

**@Pipe**

**Etc**

**Angular created lot of pre-defined decorator which help to make the class is type of special class.**

**Component control the view or part of view of web page.**

**@Component**

**selector : “app-root” : it is consider as user-defined tags. <app-root></app-root>**

**Using Angular we are creating user-defined tags with help of @Component decorator.**

**P, div, h1 to h6.**

**<p>Welcome to HTML Page</p> pre-defined**

**<myTag></myTag> user-defined**

**templateUrl : This property connect to HTML code or html page.**

**So wherever we use selector as a user-defined tags the content present in html page it display.**

**styleUrls: It is use to connected the external css file like link tag in normal html page with external CSS page. It is optional.**

**Now open app.module.ts file**

**Angular use @NgModule decorator to make this class is a type of module class.**

**According angular module is a collection of more than one component and more.**

**Properties**

**declaration : all component declaration must be provide inside this properties.**

**Import : this properties is use to import pre-defined as well as user-defined modules.**

**browserModule : this module is use to display the content on web page.**

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

**Bootstrap : This properties is use to provide parent component to load as when angular application start.**

**Create two new project (don’t create nested projects). (outside demo-app).**

**ng new angular-data-binding**

**ng new angular-forms**

**no routing and styling css**

**so we create two new apps besides demo-app(not side).**

**MEAN Stack**

**Phase 2**

**Day 4**

**11-08-2021**

**To run the project**

**ng serve –o**

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

**Syntax**

**ng generate component componentName**

**or**

**ng g c componentName**

**Data binding : Data binding is use to share the data between html (template) to component (typescript file) and vice-versa. It provide bridge between component to template.**

**2 types**

1. **One way data binding** 
   1. **String interpolation -🡪**

**Component ------------------🡪 View or Template**

**{{}}**

**Syntax**

**{{variableName}} {{name}}**

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

**{{functionCall}} {{sayHello()}}**

* 1. **Property binding**

**Component -----------------🡪View or template**

**[]**

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

**<input type=”text” value=”lname”/> in html**

**<input type=”text” [value]=”lname”/> in angular it is known property binding.**

**Angular search variable name as lname inside a component and that value display inside a text fields.**

**<div [innerText]=”lname”></div>**

**<p [innerText]=”lname”></p>**

**<span [innerText]=”lname”></span>**

**In String interpolation everything consider as string type.**

**But in property binding we can use other data type rather than string also.**

**If you want to display only output please use string interpolation. If you want to do any dynamic dom operation we can use property binding.**

* 1. **Event binding :**

**Template ------------------🡪 Component**

**()**

**Angular use same event provided by JavaScript. Only different angular remove pre-fix ie on keyword and event name wrap with ().**

**JavaScript event Angular Event**

**onClick (click)**

**onDblClick (dblclick)**

**onMouseOver (mouseOver)**

**onSubmit (ngSubmit)**

**onChange**

**onLoad**

**onUnload etc**

**using angular event we call Typscript function which is part of component without creating the object of that class.**

**using event we can change the properties value in components.**

**Template reference : Angular provide template reference which help to pass the value of textfield, password field, radio button etc to component**

**Syntax**

**<input type=”text” #referenceName/>**

**We can achieve two way data binding with combination of event binding and string interpolation or property binding.**

**Event binding**

**Template -------------------🡪Component**

**String interpolation or property binding**

**Component ------------------------🡪Template**

1. **Two way data binding**

**In template if we do any change it update in component and vice-versa.**

**[()]**

**ngModel is use to achieve two way data binding.**

**Syntax**

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

**ngModel is a pre-defined attribute part of FormsModule. So we have to import FormsModule inside app.module.ts in import section.**

**Angular Forms**

**If we want pass the value from template to component we are using template reference.**

**Using template reference more than one dom reference become more complex. To overcome this problem angular provided forms features.**

**So using Angular forms we can pass more than one value inside a container (reference).**

**Angular support two types of forms**

1. **Template Driven Form : template -----------> Component**

**Good for simple type of forms.**

**More coding on template or html side.**

1. **Model Driven Form or reactive forms. Component -------🡪 Template**

**Good for complex type for forms.**

**More code on component side.**

**Login page.**

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

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

**In Template Driven form we have to create the form reference.**

**<form #loginRef=”ngForm”>**

**</form>**

**ngForm is a pre-defined attribute part of FormsModule. So we have to import FormsModule in app.module.ts file inside a import sections.**

**MEAN Stack**

**Phase 2**

**Day 5**

**12-08-2021**

**In model Driven form first we have write code in Ts file.**

**According model driven form TextField, PasswordField, radiobutton, checkbox etc are consider as FormControl. So if you want to create any FormControl is must be inside a FormGroup.**

**FormGroup is a collection of more than one FormControl.**

**Angular provide pre-defined API ie FormGroup and FormControl**

**Login Page FormGroup**

**UserName TextField FormControl**

**Password PasswordField FormControl**

**Submit Reset**

**In Reactive form or model driven form we have to create the reference of FormGroup and FormControl**

loginRef = new FormGroup({

    user:new FormControl(),

    pass:new FormControl()

  })

**FormGroup and FormControl is a part of @angular/forms so we have to import it**

import { FormControl, FormGroup } from '@angular/forms';

**Then these reference we have to use in template**

<div>

    <h2>Login Page using Model Driven Form</h2>

    <form [formGroup]="loginRef">

        <label>UserName</label>

        <input type="text" formControlName="user"/><br/>

        <label>Password</label>

        <input type="password" formControlName="pass"/><br/>

    </form>

</div>

**formGroup and formControlName is a pre-defined attribute part of ReactiveFormsModule.**

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

**Form Validation :**

**Without writing all forms fields when user click on submit button the request will pass to server side technologies in HTML and JavaScript.**

**But In Angular we are sending the data to component.**

**Angular Forms Validation**

**Angular provided set of pre-defined classes to do the form validation**

**ng-valid**

**ng-invalid**

**ng-touched**

**ng-untouched**

**ng-dirty**

**ng-pristine**

**State class if true class if false**

**Control value is valid ng-valid ng-invalid**

**Control value has ng-dirty ng-pristine**

**Changed**

**Control has been ng-touched ng-untouched**

**Visited**

**ng new types-of-directives**

**Angular Directives : Using Angular directives we can add extra behaviour or functionality to DOM(HTML code).**

**3 types of directives**

1. **Component directive: Using this directive we are creating user-defined tag with help of selector. The component connect to html page using templateUrl. The template page contains static as we as well as dynamic data with the help of data binding.**

**@Componnet({**

**selector:”my-tag”,**

**templateUrl:”./filename.html”**

**})**

**class MyComponnet {**

**msg:string=””;**

**}**

1. **Structure directive : using structure directive we can add or remove dom elements from html page.**

**\*ngIf**

**\*ngFor**

1. **Attribute directive : using attribute directive we can apply dynamic styling for web page**

**ngStyle inline css**

**ngClass external css**

**now move inside a project folder**

**cd types-of-directives**

**ng g c structure-directive**

**ng g c attribute-directive**

**conditional statement in template**

**<div \*ngIf=”boolenValue”>**

**</div>**

**Looping in template**

**<div \*ngFor=”let variableName of arrayName”>**

**<p>{{variableName}}</p>**

**</div>**

**Please create new project**

**ng new angular-service**

**routing : no**

**styling : css**

**MEAN Stack**

**Phase 2**

**Day 6**

**13-08-2021**

**Angular service**

**If we write any logic (business logic) inside a component it may be simple or complex that code local to that components.**

**Same logic we can’t access in another components**

**Template component class service class**

**First-component.html first-component.ts**

**Variables and function**

**Business logic**

**Second-component.html second-component.ts**

**Variables and functions**

**Using angular service we can achieve separation of concern. The business logic written in service class globally can access by all components.**

**Angular Service divided into two types**

1. **User-defined service** 
   1. **Create service class object using new keyword.**
   2. **Creating service class object using DI (Dependency Injection).**
2. **Pre-defined service**

**Open angular-service project**

**Create two components**

**ng g c first**

**ng g c second**

**MVC : Model View Controller**

**Service Template Component**

**Component is a intermediate between template and service.**

**IOC : Inversion of control :**

**It is design pattern. In place creating and maintaining any resources. Allow to create and maintain by container(container is a engine part of server etc). pull from container whenever you required.**

**Rather than creating object of any class explicitly allow to create by container and pull from container.**

**IOC is concept.**

**DI: Dependency Injection**

**DI is a implementation of IOC.**

**We can achieve DI 3 ways**

1. **Setter base DI**
2. **Constructor Base DI**
3. **Interface base DI**

**In Angular We can achieve DI using constructor base.**

**If we want to create Service class using DI concept**

1. **First we have to create user-defined class with decorator @Injectable**
2. **Service class details we have to provide in provider attribute in app.modue.ts file.**
3. **Then we have to achieve DI using constructor in every component if require.**

**All component details you have to provide in declarations properties in app.module.ts file**

**And all service class details we have to provide in providers attribute in app.module.ts file.**

**Pre-defined service**

**Angular provided pre-defined API ie HttpClient. Using HttpClient we can call backend technologies REST API Service develop in any language.**

**HttpClient allow to call all Http protocol methods like get(), post(), put() and delete() etc.**

**RxJS : Reactive JavaScript**

**Steps to call REST API using HttpClient**

1. **We have to do DI for HttpClient in side a service class using constructor.**
2. **HttpClient API is a part of HttpClientModule. So we have to import HttpClientModule in app.module.ts file insider a imports section.**

**Observable Vs Promise**

**HttpClient all method(get,post,put,delete) return type is Observable. So if you want to load the data we have to use subscribe() function.**

**subscribe() function takes 3 parameter as a callback function.**

**1st parameter is next() to load the data one by one from REST API. like then in promise**

**2nd parameter is error() if any error generate at beginning or middle or end this function get call. Like catch in promise**

**3rd parameter completed. After loaded all data successfully third parameter get call.**

**Create new component**

**ng g c fake**

**So we have to create model class / interface which use to map the json data.**

**MEAN Stack**

**Phase 2**

**Day 7**

**13-08-2021**

**ng new component-communication**

**routing –no**

**styling –css**

**Every component connected through html page**

**If component contains any variable (may be number, string or array) we can access those properties within that component or that component’s html page.**

**But some if we want to share the value from one component to another component we have to check the relationship between two component and using some technique we can share the value between two components.**

1. **Parent – Child Relationship --🡪 with help of @Input decorator we can share the value from parent to child component.**
2. **Child – parent relationship 🡪 with help of @Output and EventEmitter we can share the value from child to parent. Or @ViewChild we can access child properties and behaviour in parent component.**
3. **Sibling relationship 🡪 sessionStorage or localStorage part html5 and JavaScript.**

**With help of service class with @Injectable decorator.**

****

**In new project create two component**

**ng g c child1**

**ng g c child2**

**run the application**

**ng serve –o**

**ng g c child3**

**ng g c child4**

**creating service class object using ng command**

**ng g s shared**

create new project

ng new angular-routing

routing -🡪 Yes

style 🡪 css