Day 1

15-06-2021

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

http : protocol , hyper text transfer protocol

s : secure

www: world wide web

google : domain

com : commercial

req(http/https---🡪

Client Server

🡨--Res(http/https) HTML/HTML5

CSS/CSS3

HTML : Hyper text mark up language. JS(JavaScript)

HTML : it is use to display the content on browser

CSS : apply good look and feel or presentation logic on contents.

JS : Action on contents or events on contents or programming on web page.

VS Code

Html contains lot of pre-defined tags

Syntax

<tagName>

</tagName>

<tagname/> self closing tag

1, 2 , 3 HTML4

<!doctype html public =”url…..dtd”>

Document type definition

This file contains rules or structure of html page means that is root tag name, how many child tag it must contains, number of tags inside body etc.

HTML5

<!doctype HTML> optional

Heading tags

H1 to h6 tags

Attribute : attribute is known as properties of tags.

Every tag contains one or more than one attribute. Attribute in the form of key-value pairs we have to use attribute in opening tags.

<tagName key1=”value1” key2=’value2’ key3=value3>

</tagname>

Hyperlink tags

Using hyperlink we can connect more than one web page.

1. External hyperlink
2. Internal hyperlink or bookmark

External hyperlink

<a href=”tagetPageName.html”>Text</a>

Internal Hyperlink or bookmark

One Page application (HTML).

Images :

<img src=”nameoftheimage.format”/>

List tags

Ul

Ol

Table tags

Employee Details

**Id Name Salary**

100 Raj 12000

101 Seeta 14000

Forms tags

Before HTML 5

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

After HTML 5

<input type=”email/url/search/date/date-time/time/color”/>

By default every html page form method consider as a GET method.

If method is GET the information send through URL using URL rewriting technique.

URL?key1=value&key=value&key=value….

<file:///C:/Users/91990/Desktop/MEAN%20Stack/MEAN%20Stack/Programs/HTML%20and%20HTML5/home.html?uname=Ravi&pname=123>.

If method is GET. Data is not secure.

If we want to data secure the new have to use method as POST. If method is post the data will send through body part of request.

Body method after click on submit button

<file:///C:/Users/91990/Desktop/MEAN%20Stack/MEAN%20Stack/Programs/HTML%20and%20HTML5/home.html>

Performance wise get is faster than post. Using get we can send only 255 character data.

REST API

Get

Post

Put

Delete

But html or html5 support only get or post.

CSS : Cascading style sheet.

CSS provide lot of pre-defined attribute which help to apply good look and feel (presentation logic) to web page.

Using CSS we can achieve separation on concern.

Actual content Formatting style

Type of CSS

3 types

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

Inline CSS

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

</tagname>

Internal or Embedded CSS

Syntax

This tag must be inside head tag.

<style type=”text/css”> tag

selector {property : value}

</style>

Types of selectors

1. Universal selector : \* {property : value}
2. Specific selector : tagName {property : value}
3. Multi selector : tagName, tagName { property : value}
4. Local class selector : tagName.className{} {property : value}
5. Global class selector : .className {}
6. Id selector :

Class : group of tags.

Id : using id we can make the tag is unique.

<div>

<p class=”abc” id=”p1”>First Para</p>

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

<p class=”abc” id=”p3”>Third Para</p>

<p class=”xyz” id=”p4”>Fourth Para</p>

</div>

Box model

In JavaScript all html tag is know as DOM (Document Object Model).

DOM elements : it may be html, head, body, p, div etc.

According to box model every dom element follow box model concept.



Day 2

16-06-2021

JavaScript was object based interpreter scripting language.

Object based vs object oriented

Interpreter Vs Compiler

Scripting vs programming

OOPs : class, object, EPIA

JavaScript using ES5. It provide lot of pre-defined object as well as we can create user-defined objects.

We can write JavaScript code (internal or external JS file).

Syntax of script tag

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

</script> closing tag

This script tag we have to write in between head tag, body tag or without any tags. But extension of file must be .html

Variable : variable is name which hold some value.

Syntax to declare the variable

var variableName;

data type : java script support loosely type data type.

Operator :

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

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

Logical operator : &&, ||, !

Assignment operator : =

Increment and decmrement : ++, --

Ternary operator : condition ? true: false;

Type of operator : typeof

If statement

1. Simple if

if(condition) {

}

1. If else

if(condition) {

}else {

}

1. Nested if

if(condition) {

if(condition) {

}else {

}

}else {

}

1. If else if or ladder if

if(condition) {

}else if(condition) {

}else if(condition) {

}else {

}

switch:

syntax

switch(variableName) {

case label: block1;

break;

case label: block2;

break;

case label: block3;

break;

case label: block4;

break;

default : wrong block;

break;

}

Looping :

While loop

var i=0,n=10;

while(i<=n) { entry loop

do task

i++ or n--

}

Do while loop

var i=,n=10;

do {

do task

i++ or n—

}while(i<=n); exit loop

For loop

Syntax for loop

1 2 4

for(initiliazation;condition;increment/decrement) {

3

}

for(var i=0;i<=10;i++) {

document.write(“i ”+i+”<br/>”)

}

function : set of instruction to perform a specific task.

JavaScript function are divided into 2 types

1. Pre-defined global function
2. User-defined function

Pre-defined functions.

1. alert(“Msg”): it is use to display the alert message.
2. prompt(): this function is use to receive the value through keyboards.
3. parseInt(): it convert string to integer
4. parseFloat(); it convert string to float
5. eval(): it covert string to number.

do {

using alert display option

1: add, 2 : sub

switch()

case 1 Add display sum

case 2 Sub display sub

do you want to continue

}while()

Thank you

User-defined functions

Normal function style

function functionName(parameterList) {

function body;

}

1. function no passing parameter and no return type.
2. Passing value but no return type.
3. Function passing parameter and return value.
4. Function no passing parameter but return value.

Event : event is provide the interaction between user and component(dom elements).

Event provide bridge between html and JavaScript code.

JavaScript or DOM elements

JavaScript all event start with on prefix followed by name of the events.

onClick

onDblClick : button or any tags.

onMouseOver : image or map

onMouseOut

onKeyUp : textfield, password field.

onKeyDown

onChange : dropdown

onFocus : enter inside text field

onBlur : exit from text field.

onSubmit : validation with submit button

onLoad : body tag page loaded

onUnLoad : page refresh or close the application

etc

Listener : Listener are normal JavaScript function which help to listen the generated events.

**DOM : Document Object Model**

index.html

<html>

<head>

<title>Message</title>

</head>

<body>

<h1>Heading Message</h1>

<p>Paragraph Tag</p>

</body>

</html>

In browser memory it create DOM hierarchy

Html -🡪 root tag (Node )

Head body

Title h1 p

textNode – Message Heading

Paragraph Message

DOM API (Document Object Model) Application Programming interface.

Java, C,Python, C#, JavaScript provide pre-defined function or classes which help to read, write and update dom dynamically.

JavaScript DOM API

1st way to retrieve the value of text field.

document.formName.textfieldName.value

2nd way to retrieve the value of text field

document.getElementsByName(“textFieldName”)[index].value

We can write more than one tag have same name attribute.

3rd way to retrieve the value of text field

document.getElementById(“idName”).value

Form Validation

Using JavaScript

Using HTML5 features

Day 3

17-06-2021

object : object is any real world entity.

Properties or state have variables/fields/state

Person

Behaviour do/does functions/ methods

Bank

Animal

Car

Phone

Till ES5 there is no class keyword.

2 type of objects

1. User-defined object
2. Pre-defined or built in objects.

Using function we are creating user-defined objects.

Class : it a blue print of object or template of objects or use-defined data type which help to describe the objects.

Pre-defined objects

Java Script follow object hierarchy

object -🡪 property

behaviour

object -🡪 property

behaviour

object -🡪 property

behaviour

object

BOM Hierarchy : Browser Object Model

window : window is a top most object in BOM hierarchy

DOM Hierarchy : Document Object Model

Window object

document

alert()

prompt()

confirm()

open()

close()

setTimeout()

setInterval()

clearInterval()

asynchronous and synchronous communication

synchronous execution of statement

document.write(“1st statement”);

document.write(“2nd statement”);

document.write(“3rd statement”);

asynchronous execution of statement

asyn document.write(“1st statement”);

document.write(“2nd statement”);

document.write(“3rd statement”);

synchronous function call

fun1()

fun2()

fun3()

fun4()

asynchronous function call

fun1()

asyn fun2()

fun3()

fun4()

client server synchronous communication

asyn 1st Req

asyn 2nd Req

3rd Req

4th Req

Client Server

setTimeout

setInterval

clearInterval

These three function part of window object which help to achieve asynchronous operation on client side or view side.

CSS property using DOM CSS property

color document.getElement\*(“”).style.color

font-size document.getElement\*().style.fontSize

Basic Pre-defined objects

Date : This object is use to display date and time.

Day 4

18-06-2021

DOM Hierarchy

Index.html

<html>

<head>

</head>

<body>

<p>Welcome to <b>HTML</b> Web Page</p>

<font color =”” size =5 face=”arial”>

</font>

<div>

</div>

</body>

</html>

DOM API : Creating, Delete DOM element dynamically.

ES : ECMA Script European Computer Manufacture Association.

Programming concept

OOPs concept

ECMA is a concept.

JavaScript is one of the implementation of ES or ECMA.

ES5 Version

ES6 or ES2015

JavaScript using ES6 features.

Till ES5 to declare the variable we are using var keywords.

But from ES6 JavaScript we can declare the variable using let, var and const keywords.

var : using var keyword we can re-declare same variable once again with same vale or different type of values.

var a=10;

a=20; // Re-initialization

var a = 30; // re-declaration

int a=10;

a=20;

int a=20; //Error

using let keyword we can’t re-declare same variable once again with same value or different values.

Using var we can declare global scope.

Using let we can declare local or block scope.

const : it use to declare constant value.

Types of functions

function :

1. Normal function :
2. Expression style function :

If function is normal function declaration we can call that function before function declaration.

Expression style function we can’t call before function declaration.

Arrow function : arrow function is a replacement of anonymous function style.

Arrow function by default return the value without return keyword.

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

Basic or utility JavaScript objects.

Array :

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

In JavaScript array is not a fixed memory size.

Syntax to declare the array

Literal style

let variableName = []; empty array

let variableName = [100,200,300,400,500];

memory creation style

let variableName = new Array() length 0

let variableName = new Array(10) length 10

let variableName = new Array(100,200,300,400)length 4

normal loop

for in loop

for of loop

forEach() : forEach() function takes callback function as a parameter(may be normal function, expression style function or arrow function) which help display the value one by one.

splice() : using this method we can add, remove and update elements in between of array.

Splice(indexPosition,deleteCount,updateElement…..)

Day 5

21-06-2021

sort() methods

created user-defined object using function style

function Employee() {

this.id=100;

this.name “Ravi”;

this.dis= function() {

}

}

let emp1 = new Employee();

document.write(emp1.id);

emp1.dis();

create user-defined object literal style.

Rest and spread operator : …

Rest and spread operator with array

Rest operator is use to store zero,1 or many array value while de structure of array.

…variableName

Spread operator with array

Rest and spread operator with function

JSON

JavaScript Object Notation

Req Java(req)

Client HDFC XML/JSON SBI

Res .net(Res) Asp.net

**Java**

**Spring boot**

Asp.net

Php

Python

Node JS

Web Service :

**JavaScript Object Notation (JSON)**

Json store the data in the key value formats.

Key must be unique and type must be string.

Vale may be number, Boolean, string, array, array of objects

{“key1”:valu1,”key2”:”value2”,”key3”:value3}

Json Conversion methods.

Converting to string to json

    let empInfo = JSON.stringify(emp);

// string to json

    let jsonData1 = JSON.parse(empInfo);

Promise :

Promise is a pre-defined object part of JavaScript which help to handle asynchronous events of data. Data may be string format, xml format, json format etc.

Promise resolve or rejected depending upon the conditions.

Full fill or rejected or pending.

User-defined promise

Pre-defined promise

To retrieve the data from promise object we have use then() and catch() callback functions.

If promise resolved then() call back function get called which help to read data from promise.

If promise rejected catch() call back function get called which help to read error message.

fetch() : It is a pre-defined function provided by JavaScript which help to consume REST API.

Return type of fetch() method is promise objects.

1st Task

Display fake json data in proper table format.

2nd Task

TextField -🡪 1 keyup events call function and take the value of textfield using dom API and append that value through url and display that records details in same page below textfields.

If id correct display records or display no record founds.



ES6 OOPs Concept.

class :

constructor : constructor is a type of special function which help to create the objects.

While creating the constructor we have to write the function name as constructor. Constructor get called automatically when we create the object.

Polymorphism

Compile time : JavaScript ES5 or ES6 doesn’t support

Run time : function overriding.

The function have same name and same signature is known as function overriding.

Day 6

22-06-2021

Scope objects : scope object is use to share the data between more than one js pages.

sessionStorage :

localStorage :

setAttribute(“key”,value);

getAttribute(“key”);

removeAttribute(“key”);

sessionStorage store the value till the application open. Once application close the value from sessionStorage get destroy.

localStorage store the value permanently.

async and await keywords are part of ES7 features.

It is use to handle promise data.

ES5 and ES6

fetch(“url”).then().then().then().catch()

TypeScript : TypeScript is super set of JavaScript. Typescript is one of the implementation of ES6 features.

TypeScript support data types concept.

Browser can’t understand or we can’t use typescript file (ts) file in html page directly.

So we have to convert TS to JS then include generated js file in html or view page.

Transpiler : It is type of compiler which help to convert one format of file into another format of file.

typescript : tsc : converting ts to js : Angular

babel : JSX : converting JSX to Plain JavaScript

want to convert ES6 to ES5.

jQuery : jQuery external JavaScript library which contains lot of pre-defined function which internally connected to each other to read, write and update DOM element easily.

Few are library and few are framework.

Ext JS

Backbone JS

Coffee JS

React JS

Angular

Vue JS

Node JS : Node JS is not a library or not a framework.

Node JS is a run time environment for the JavaScript program + library or Framework.

Before Node JS JavaScript use for Client Side scripting language.

Frontend backend

HTML,CSS,JavaScript Java JEE

jQuery Servlet/JSP

Spring boot

Asp.net

Php

Python

Node JS

After Node JS JavaScript we can client side as well as server side scripting language.

Node JS provide lot of pre-defined modules. Those module may be local module, external module which help to create server side programming language using JavaScript.

Before Node JS if we want to run any JavaScript code

1. We have to write script tag inside a html code
2. We have to create external JS file and include in html page.

After node we can run External JavaScript program using command prompt.

Node JS : mainly use to create the Server side programing language.

In Node JS script program we can’t use window and document objects.

To convert ts to js we require tsc

npm : node package manager. Npm command is use to download external dependencies or modules written in JavaScript.

npm is like mvn(Maven) in Java.

Syntax

npm install –g moduleName : globally

npm install moduleName : locally

install typescript external module using npm command.

npm install –g typescript

TypeScript features.

1. let, const and var
2. dataTypes : typescript support strict data types concept.

Syntax data type

let variablename:datatype;

let a:number=100;

let b:number=100.20;

let name:string =”Ravi Kumar”;

let result:boolean = true;

let num = 100; // valid

num =”Ravi”;

let id:number;

let fname:string;

let age:number;

let msg:any;

msg=100;

msg = “Ajay”;

msg = true;

1. array with data types.

ES5 or ES6 to declare array variable

let/var variableName = 100;

let/var variableName = [100,200,300,400,500]

let/var varibleName;

array can hold any type of values.

Function

1. function with number of parameter must be match
2. function with number of parameter as well as type of parameter must be match.
3. Function with specific return type or no return type or generic (any) return type.
4. Function with optional parameter. Optional parameter must be last parameter in functions if you are using only one variable as optional parameter. If we want more than one optional parameter so we have declare from right to left but no gap between two variables.
5. Function with default initialization.
6. Rest operator and spread operator with function

Oops Concept in typescript using ES6

object

class

Encapsulation : binding or wrapping data (variable) and code (function) in a single unit.

class

access specifiers

**public :** if variable or function are public we can access those variable and function through object of that class.

**private** : if variable private we can’t access those variable outside class directly as well as with the help of objects.

**Constructor short cut initialization.**

**Object : few methods**

**Inheritance :**

1. Single inheritance : one super class and one sub class
2. Multilevel inheritance : one super class and n number of sub classes one by one
3. Hierarchical inheritance : one super class and n number of sub class directly connected to super class.
4. Typescript doesn’t support multiple inheritance using classes but it can support using interface.

**interface**

interface interfaceName {

id:number;

name:string; //normal variable

dis():void; // incomplete functions

}

**Module :** module is a collection of variable, function, classes or interfaces.

Module is like a package in Java.

According to typescript .ts file itself is known as module.

Import and export keywords.

Typescript configuration **tsconfig.json**

tsconig.json hold all configuration details for you application.

To create the configuration file using command we have to run the command as

**tsc –init**

Set

Map

PName

DOB : type=”date”

Group

Blood group

Blood grouse reading

<65 below

65 to 100 normal

100 > above

****

Day 7

24-06-2021

Webpack : webpack is a static module bundle webpack teats all files and assets as modules.

Using webpack we can create dependencies graphs which describe how modules are related to each other using import (require) or export.

npm install –g webpack

npm install –g webpack-cli

after installation we have to create static bundle file.

**webpack.config.js**

**git :**

**github**

Sub Version control tools :

Version Control system : Version control system that records changes on files or folder and projects or application.

Local Version Control : RCS : Revision control system.

Centralized version control :SVN

Remote repository

Team 1 Team 2 Team 3

Distributed Version control system : GIT

Remote repository

Local rep local rep local rep

Team1 Team2 Team3

Git is type of Distribution version control which keep tracks of software version and allow many developers and team to work on given projects within a maintaining connection of common network.

This make only folder as local repository

git init

git status

git add : This command is use to add the file in staging area.

**Git** : it is software which provided set of commands which help to maintain the local repository as well as interact with remote repository.

Github : remote repository

AWS : code commit

Azure

Oracle cloud

Etc

How to create local repository from existing remote repository

**git clone url :** first time fresh copy of the existing

repository.

**git pull :** to get the updated content from remote repository.

Git is sub version control system

**Git branch :**

Git branch is like pointer which hold more than one commit or snap short of our changes.

By default branch is master or main

View all branches or default branch name

**Git branch**

To create new user-defined branch

**git branch branchName**

command to switch the branch

**git checkout branchName**

creating as well switch to user- defined branch

**git checkout –b branchName**

25-06-2021

.divTag {

color:red;

font-size:24px;

}

.fontClass {

}

.pTagClass {

}

Bootstrap : bootstrap is a open source CSS framework for web application as well as mobile application which help to create responsive web application.

Bootstrap provide lot of pre-defined css classes respective dom elements.

Div, p, table, button, form etc.

**Bootstrap classes**

Container and container-fluid

Container : it known as fixed width container. It take full with for web page.

Container-fluid : it leave left and right side space.

<div id=”footer”>

</div>

<div id=”nav”>

</div>

<div id=”aside”>

</div>

<div id=”main”>

</div>

<div id=”footer”>

</div>

Html5 sematic tags

<header>

</header>

<nav>

</nav>

<aside>

</aside>

<footer>

</footer>

**Bootstrap grid layout**

Grid layout allow us display the dom elements up to 12 column across the page.

You and divided the container/container-fluid in row and columns. By default each row consider as 12 columns.

Screen size

xs <576px

sm >=576px

md >=768px

large >=992px

xl >=1200px

**DOM API :**

jQuery is a external JavaScript library function which contains lot pre-defined function which internally connected to each to do Dom operation very easily.

Once the document loaded

$(“select dom using tagname,id selector,class selector”).doActionOnTag();

document.getElementByTagName(“\*”).length

Library as well as framework.

jQuery

Coffee js

Backbone js

Ext JS

Angular JS

Angular Framework

React JS

Vue JS

Office JS

**Node JS**

Node JS is not a library or not a framework.

Node JS run time environment for JavaScript library or Framework.

Java -🡪JRE

JavaScript 🡪 Node Js

Front end Server Side Technologies

HTML Servlet, JSP and EJB

CSS Spring Framework and

JavaScript Boot

Bootstrap Asp.net

jQuery Php

Phython web application

Node JS

After Node JS we can use JavaScript to develop to Server side technologies.

“Node’s goal is provide an easy way to build scalable network programming”.

Javascript callback and asynchronous operation

Node JS is a open source, platform independent or cross platform which help to developer server side as well networking application.

It is written use JavaScript. Application level.

It provide an event driven architecture and non blocking io operation. Node js achieve the features as optimize and scalability.

Node JS use Google JavaScript V8 engine to execute the node JS Programs.

Unique in Node

1. After node JS JavaScript we can client side as well as server side.
2. All server are thread base but node js a event base architecture.
3. Node JS running on V8 engine.
4. It is not library or not a framework.
5. Node JS is not a multi threaded.

Node JS we can’t access DOM and BOM hierarchy.

Node JS is not use to create GUI application.

Node is a platform for writing JavaScript application outside browser.

REPL terminal : Read Eval Print Loop

Global objects

console

process

**Node JS Modules**

Modules is a simple or complex functionality organized in single or multiple JavaScript files which can be reused through the Node application.

Types of Modules

Core Modules

fs

os

path

url

http

etc

var referenceName = require(“moduleName”);

load the module

let

Local module (user-defined modules)

Third party modules

**fs module** fs is a core module which is use to do the synchronous as well as asynchronous file handling operation.

readline : it is a external module which help to take the value through keyboards. readline module is type of async module.

Npm install –g moduleName globally

npm install moduleName locally

readline-sync : It is use to take the value through keyword as synchronous manner.

npm install readline-sync

URL :

url module provide the URL details in server side.

<http://localhost:9090?name=Ravi&salary=12000>

action =”home.html”

localPath:home.html?user=Ravi&pass=123

when we fill the form if method is GET it use query param concept.

url?key=value&key=value&key=value

**http** : http is a pre-defined core module which help to create server using JavaScript. Using this module we are going to create web page on server side technologies like in Java Servlet/JSP, Aps.net or php or python etc.

all non-node js server are thread base

Tomcat -------------🡪 Servlet/JSP

1,000 etc client request can handle at same time.

1st client send request to server --------🡪1 thread

All 1,000 client send the request and doing some task in that application.

Thread base server are lock or block the request.

Node JS Server are use even driven architecture.

class Booking {

int avl=1;

}

Booking b1 = new Booking();

Thread t1

Thread t2

Thread t3



**29-06-2021**

Node JS third party web framework.

There are various third party open source framework available in node package manager (NPM). Which makes node.js application development fast and easy.

Express

Koa

Hapi.js

Geddy

Etc

npm install –g express

npm install express

configuration file

This file hold all project configuration details.

package.json

npm init : it will ask few options

npm init –y : automatically create the file with default details.

Express module provide get, post, put and delete http protocol methods to handle the different type of request.

View technologies is : Plain HTML

**Login.html**

Get and Post

Get method : we can call through browser, through hyperlink or button or form submit button

Post method : form submit button

In Node JS with Express module we have to use body-parser third party module which help to unable body part data from request.

npm install body-parser

**Mini Bank Application and bootstap**

**Validation HTML5 or JavaScript.**

Express JS provide template : this template like a html we can use programming.

Default : <http://localhost:9090>

//sessionstorage and localStorage

Login Page

Post

EmailId :

Password :

Sign In Reset

Hyperlink : SignUp

Post

FirstName :

LastName :

Gender :

PhNumber:

EmailId : email: unique

Password :

Address : TextArea

Account Create Reset

fs array set map

After account created successfully now we can do login.

Dashboards

Hyperlink

Get methods

Check Balance withdraw deposit

0 or 1000 min 500 maintain 50,000 max

Before web service all view (HTML,CSS and JavaScript) are tightly coupled with backend technologies.

NodeJS

JEE(Spring boot)

Asp.net

Php etc

Client Server Server

HDFC HSBC

Java asp.net

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

Frontend backend

HTML,CSS,JavaScript REST API Node JS

and Typescript

Angular

Entity : customer, person, employee, login, trainer, students, projects.

REST API : all entities is known as resources.

In REST API we share the object using xml or json format.

Get : get the resources : all customer details, all employee details, get customer details by id. Get account balance using id. Etc.

Select query

Post : create the resource : customer details, employee details, student details. Etc.

Insert Query

Put : update the resource : customer age or address or phone number using cid, update salary using empid etc.

Update Query

Delete : delete the resource : delete customer details using cid, employee details using empid

Delete query

30-06-2021

REST API : Representational State Transfer Application Programming interface.

Using REST API we are exposing resources as a web service.

Server.js or app.js or main\_server.js

Get() and Post() :

http : provide 9 methods.

4 methods

Get(), post(), put() and delete()

app.get(“/path,callback); URL, hyperlink or button or submit button

app.post(“/path,callback); : submit button

app.put(“/path,callback);

app.delete(“/path,callback);

According REST API we can all those using URL

Contract file

Types of Web Service

1. SOAP :Simple Object Access Protocol :SOA (Service oriented architecture).

SB

lookup Register

WSDL(XML)

call the service

SC or SR SOAP Req SP

SOAP Res only in XML format

Python Java

Asp.net

float findBalance(int accno){

return 45000;

}

Using WSDL file they are creating code respective that language which help them to call Web Service.

In SOAP web service data only in the form XML.

1. REST API :

SOAP :WSDL : Web Service Description Language.

REST API :WADL : Web Application Description Language

AJAX : Asynchronous JavaScript and XML

XMLHttpRequest and ActiveXObject

In REST web service we are not generating any code which help to call the service.

REST API data may be any form like xml, json, plain text, html or media type etc.

REST web service is style

Create the folder (express with rest api)

**npm init –y (package.json)**

**npm install express body-parser**

if we want to pass the value to REST full Web service using get methods

using 2 ways

1. Query params
2. Path params

Query param

By default HTML or HTML5 form if method is get internally they use query params concept.

URL?key=value

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

Path params

URL/value1

URL/value1/value2/value3

You have to create REST API

Product Details

Array or Set API.

1. Get all product details
2. Get product using Id if id present display product details not product not found in json format.
3. Store product details. Id must be unique and price > 1000.
4. Update product price : price only increment.
5. View all product details
6. View product details ascending order and descending order using price and name.
   1. You can create four URL
   2. You can create one generic url
7. Delete product details using Pid

Node JS and Express JS using TypeScript.

Create src folder and keep the all JavaScript file inside this folder.

Do the change in package.json file as

"main": "src/app.js",

  "scripts": {

    "start": "node .",

To run the application we have to use the command as

npm run start

development mode

npm install –D typescript

npm i --save-dev @types/node

or

npm i -D @types/node

npm i -D @types/express

now we have to create tsconfig.json using commands as

tsc –init

**do changes in tsconfig.json file**

 "outDir": "dist",

write the program

app.ts

import express,{Request,Response} from 'express';

let app = express();

app.get("/",(req:Request,res:Response)=> {

    res.send("Welcome to Express")

})

app.listen(9090,()=>console.log("Server running on port number 9090"));

then convert this program to js

**tsc**

Then do changes in package.json file

"main": "dist/app.js",

Then run the command as

npm start run

01-07-2021

TypeScript

App.js : entry file load all modules express as well as body-parser

More than one get, post, put and delete

Model classes (Employee)

Service class (All business Logic)

Database : MySQL or Oracle etc

MySQL

Oracle

Db2

Sql Server 2020, 21

We can store the data permanently

1. File system
2. Database system.

File system

1. File system not a consistence. Etc txt, doc, ppt, zip. Txt file :
2. Id,name,salary 1,Raj,12000
3. Id/naname/salary 1/Raj/12000
4. Id\_name\_salary
5. Data redundancy (duplicate records).
6. CRUD operation on file system is very complex.

Data : raw facts.

Information : meaningful data or processed data.

Database : storing the data in proper format like table(using row and columns).

DBMS : Database Management system : It is software which help to store the data in table format.

RDBMS : Relational database management system.

DBMS

12 rules Dr EF Codd’s rules

Excel sheet :

Pk

SRNo Name Salary

1

2

1

TrainerStudent

TId Tname Tech Sid SName Age

1 Raj Java 100 Seeta 21

1 Raj Java 101 Reeta 22

1 Raj Java 102 Meeta 23

Trainer

PK

TId TName Tech

1 Raj Java

2 Ravi .net

Student

PK FK

Sid Name Age TS\_ID

100 Seeta 21 1

101 Reeta 22 1

102 Meeta 23 1

102 Teeta 24 2

MySQL

Oracle

Db2

Sql Server

PostgreSQL

SQL Query language

1. DQL or select query or DRL
2. DML
3. DDL
4. TCL
5. DCL

My SQL

Create database mydb;

Use mydb;

Create table emp(id int, name varchar(10), salary float);

Oracle

create table emp(id int, name varchar(10), salary float);

Limitation of RDBMS

1. All RDBMS database are use schema concept. So we have to create table with number of columns and type of values storing in those columns.
2. In RDBM we have to store the data using SQL query as well retrieve as sql query format. So we have to convert into json format manually.

Structure format

Semi structure : json, xml, other format

Non structure

No SQL Database

Key-value : redis

Graph database : Neo4j

Document Oriented : Mongo DB

Column family : Hbase , Cassandra

**Mongo DB :**

Mongo DB is a open source database which help to store the data and information in the form of documents. Mongo DB is schema less.

In Mongo DB table is known as collection(folder)

In Mongo DB record is known as document(txt,doc,pdf,subfolder).

In mongo db all document are independent.

In Mongo DB every document hold one or more than one field which independent compare to another documents.

Employee

Id Name Salary age City

100 Ravi 12000 null null

101 Ramesh 14000 null null

102 Rakesh 16000 null null

104 Lokesh null 21 null

105 Mahesh null null Bangalore

Mongo DB Database

mongod : to start the server

mongo : to run the mongo terminal

to display all databases

show databases

show dbs

my sql

create databases databaseName

use databasesName;

show database

Mongo DB

use databaseName

This command is use to create and switch to that database.

To create the collection

db.createCollection(“Sample”);

to view collections we can run the command as

show collections

show tables

In mongo db database we store the record in document in the form of json.

**Insert the document in collection**

db.Sample.insert({name:"Ravi",age:21});

In Mongo DB \_id is a pre-defined property which help to maintain the unique ness between to documents.

By default unique value created using ObjectId but if we want to we can insert the custom value and value must be unique.

Mongo db database case sensitive

db.Employee.insert({\_id:”100”,name:”Ramesh”,age:21});

View documents in different ways

We can retrieve specific document using index position.

db.Employee.find()[1];

we can retrieve specific field value in specific index position

db.Employee.find()[1].\_id;

db.Employee.find()[1].name;

we can limit and skip number of document to view

db.Employee.find().limit(2)

db.Employee.find().skip(2)

display specific field value from all documents.

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

db.Employee.find({},{name:1,\_id:0});

db.Employee.find({},{name:1,\_id:0,age:1});

products.filter(p=>p.length>5).forEach(v=>console.log(v))

02-07-2021

Create **package.json** file

**npm install express**

tsconfig.json file create

**tsc -init**

**npm install –D typescript**

npm i --D @types/express

npm i --D @types/node

package.json file

"main": "dist/app.js",

  "scripts": {

    "start": "node .",

    "test": "echo \"Error: no test specified\" && exit 1"

  },

Run the project using command as

**npm run start**

filter the document using conditions.

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

db.Employee.find({name:'Ajay'});

relational operator

db.Employee.find({age:{$gt:22}});

$lt

$gte

$lte

$eq

$ne

$and and $or

db.Employee.find({$and:[{name:'Ramesh'},{age:21}]});

db.Employee.find({$and:[{name:'Ramesh'},{age:{$gt:22}}]});

db.Employee.find({$or:[{name:'Ramesh'},{age:{$gt:22}}]});

condition and filter the fields

db.Employee.find({age:21},{name:1,\_id:0});

db.Employee.find({age:{$gt:23}},{name:1,\_id:0});

**sort with specific property**

update using conditions.

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

db.Employee.update({\_id:103},{$set:{age:32,name:'Mahesh Kumar'}});

display those document where fields present

db.Employee.find({city:{$exists:true}});

display those document fields present and sorting

db.Employee.find({city:{$exists:true}}).sort({city:-1});

db.Employee.find({city:{$exists:true}},{city:1,\_id:0}).sort({city:-1});

**Remove**

db.Emp.remove({});

db.Emp.drop();

db.Sample.remove({age:21});

it delete all document when condition satisfies

db.Sample.remove({lname:'Patil'});

it delete only one document

db.Sample.remove({name:'Raju'},1);

remove specific field using conditions

db.Sample.update({name:'Raj'},{$unset:{city:1}});

insert many documents

db.Employee.insertMany([{\_id:105,name:"Vijay",age:28},{\_id:106,name:"Balaji",age:32}]);

Express JS with typescript

Customer

Add 4 to 5 customer

Orders

Product

Add 6 products details

1. I can order one product : Order Id 112233
2. I can order more than one product : Order Id 112234
3. I want to view my order history
4. I want to view all product details.

**Collection relationship RDBMS**

**Primary key and Foreign key**

One to one Person ---- Passport PK---PK

One to many Department -🡪Employee Pk---FK

Many to one Employee -🡪 Project Pk---FK

Many to many Students ---🡪 technologies

FK-----FK

PK------🡪FK

FK🡨-----PK

In Mongo DB we can create the relationship using 2 ways

Embedding Collection

Linking Collection

Embedding Collection

Person {pid:100,pname:”Ravi”,age:21}

PanCard : {pnumber:”AABB123”,date:”30-mar-2018”}

{pid:100,pname:”Ravi”,age:21,

”panCard”:{pnumber:”AABB123”,date:”30-mar-2018”},

”add”:{“city”:”Bangalore”,”state”:”Kar”},”projects”:

[{“pid”:1122,”tech”:”Java”},

{“pid”:1123,”tech”:”Python”},

{“pid”:1124,”tech”:”Angular”}

]

}

db.Person.insert({\_id:100,pname:"Ravi",age:21,

"panCard":{pnumber:"AABB123",dob:"30-mar-2018"},

"add":{"city":"Bangalore","state":"Kar"},

"projects":

[{"pid":1122,"tech":"Java"},

{"pid":1123,"tech":"Python"},

{"pid":1124,"tech":"Angular"}

]

})

db.Person.insert({\_id:101,pname:"Ramesh",age:24,

"panCard":{pnumber:"AABB567",dob:"30-Apr-2018"},

"add":{"city":"Mumbai","state":"Mh"},

tech:["Java","Python","C","C++"]

})

**Linking style**

One to one

One to many

Student

Storing only tid

PK FK

\_id:1,SName:”Reeta”,age:21,TSId:100

\_id:2,SName:”Veeta”,age:22,TsId:100

\_id:3,SName:”Meeta”,age:23,TsId:101

\_id:4SName:”Teeta”,age:24, TsId:101

Storing trainer details.

\_id:1,SName:”Reeta”,age:21,TSId:{ \_id:100,name:”Ravi”,tech:”Java”}

\_id:2,SName:”Veeta”,age:22,TsId:{

\_id:100,name:”Ravi”,tech:”Java”

}

\_id:3,SName:”Meeta”,age:23,TsId:{

\_id:101,name:”Raju”,tech:”Python”

}

\_id:4SName:”Teeta”,age:24, TsId:{

\_id:101,name:”Raju”,tech:”Python”

}

Trainer

PK

\_id:100,name:”Ravi”,tech:”Java”

\_id:101,name:”Raju”,tech:”Python”

Store trainer details id

\_id:100,name:”Ravi”,tech:”Java”:sid:[1,2]

\_id:101,name:”Raju”,tech:”Python”,sid:[3,4]

Trainer

PK

TId TName Tech

100 Raj Java

101 Reeta Python

Student

Pk FK

Sid SName Age TSId

1 Meeta 21 null

2 Reeta 22 100

3 Veeta 23 100

4 Leeta 24 100,101

**Aggregate operator**

Aggregate operator we can use in single collection or more than collection to perform aggregation operation on it and after that it return single result to end users.

**$lookup**

**Trainer and Student2**

**Trainer and Student1**

db.Trainer.aggregate([{

$lookup:{

from :"Student1",

localField:"\_id",

foreignField: "ts\_id",

as : "Students\_Details"

}}

])

Primary collection : Trainer local field : pk of primary key of collection \_id

Secondary collection : Student ' secondary collection ts\_id like a fk.

tsid : like a fk

aggregate function with $group

db.Employees.aggregate([

{

$group: {\_id:"$city"}

}

])

db.Employees.aggregate([

{

$group: {\_id:"$city",totalSalary:{$sum:"$salary"}}

}

])

db.Employees.aggregate([

{

$group: {\_id:"$city",maxSalary:{$max:"$salary"}}

}

])

db.Employees.aggregate([

{

$group: {\_id:"$city",minSalary:{$min:"$salary"}}

}

])

db.Employees.aggregate([

{

$group: {\_id:"$city",avgSalary:{$avg:"$salary"}}

}

])

db.Employees.aggregate([

{

$group: {\_id:"$city",numberOfEmp:{$sum:1}}

}

])

**06-07-2021**

To connect the mongodb database through node js we can use two modules

mongodb module

create folder

Then create package.json file using command as

npm init –y

then install mongodb module

npm install mongodb

readline-sync

mongodb

do {

switch() {

1 : add

2 : delete

3 : update

4 : display

}

}while()

mongodb is like JDBC

Mongoose is like a ORM IN Java Technologies.

mongoose is external module which provide data modelling concept.

Using Schema we can create the Data Model.

Schema give the definition of collection ie name of collection, fieldsname as well as type of values fields must hold.

Using Schema we have to create the Model.

Using that model we can do CRUD operation on database.

mongoose module

create the folder

create the package.json file

npm install mongoose

Jade Template using Express

**MVC : Model View Controller**

View --🡪 Controller --🡪 Model

Angular or look and feel or presentation logic.

Angular

React JS

Plain JavaScript using promise object.

Controller -🡪 Intermediate between Model and View

Express JS 🡪 Controller

Get, Post, Put, Delete

Model (model class, business logic , database connectivity).

mongodb or Mongoose module

Mongo DB

View ---🡪 Browser or Browser Plugin

**app.js or Server.js**

load the express, bodyParser, mongoose

connect the database.

Route to Router base upon main path in URL.

Employee entity

<http://localhost:9090/api/employee>

Customer entity

<http://localhsot:9090/api/customer>

Run the app in port number using listener.

**Router :**

Router file check type of methods like get, post, put, and delete and sub path. Then pass the request to controller specific methods depending upon the methods.

**Controller** : retrieve model created using schema. Do the operation on that model depending upon the type of request.

**Model** : Schema help to create the collection with fields name and type of data type. Using schema we can create the Model.

Create folder

npm init –y

**npm install express mongoose**

**Angular Framework**

npm install –g @angular/cli

ng -–version to verify angular installation

create two projects

ng new welcome-app

routing 🡪No

styling 🡪css

ng new data-binding

routing 🡪No

styling 🡪 css

**HTML,CSS,JavaScript using ES5**

**Library like jQuery**

**jQuery is a external javaScript library which help read, write and update dom very easily.**

**$(“selector”).doOperation();**

**MVC : Model View and Controller**

**Java(Spring framework), Asp.net, Php, Python, Node JS**

**View Side.**

**Framework : Framework are implementation of architecture.**

**Framework is a implementation of design pattern.**

**Design pattern is best practise.**

**EOF :**

**Library is not a standard. Using Library we can do dom operation. Library is use to achieve one type of specific task.**

**But framework as standard by default.**

**Framework is for multi purpose.**

**React JS is a library in MVC react focus only on View.**

**Angular Framework Vs React JS**

**Angular JS (Framework) 1.0,1.1 1.8 etc**

**HTML,CSS,JavaScript using ES5. Angular JS framework base upon the MVC architecture.**

**Read, Write, Update DOM Operation**

**Angular JS**

**Angular Framework 2 to 12.x**

**HTML,CSS,JavaScript and TypeScript using Node JS.**

Angular Framework Version 2 to 12.x is use to create SPA (Single Page Application).

Connecting one page to another page.

Index.html home.html

Using hyperlink

Submit button

Click button with JavaScript

Using Component rather than moving from one page to another page we can replace the component by another component using some events.

Component : It always control the view or part of the view in web page.

Angular Framework

Node installation

npm (node package manager)

npm install –g @angular/cli

ng (next generation) Angular

next generation for DOM.

P, h1, form, table

Pp, mytag

Using Angular as well as react we are creating user defined tags with the help of components.

ng new project-name

After project created move inside a angular project folder.

And run the command as

npm install –g @angular/cli

ng -–version

ng new welcome-app

then move inside a project folder

cd welcome-app

ng serve

after 100% compile then open the browser and write the command as

<http://localhost:4200>

app🡪

src🡪

app.component.html

app.component.ts

Decorator : decorator is use to add extra behaviour to class or function or property. Decorator is use to provide meta-data (data about data).

In Java Decorator is known as annotation.

All decorator start with @ followed by name of the decorator.

@Component : This decorator is use to make normal TypeScript class as a component class. Component is use to control the view or part of view.

@Component decorator properties

selector : This attribute is use to provide unique id for the component. Using this selector we are creating user-defined tags.

p

h1

form

<app-root></app-root>

**templateUrl** : This property is use to connect the external html page.

**styleUrls:** This tag is use to connect external css page. It is like a link tag in html with external css page.

Open the **app.module.ts**

Module is a collection of functions, variable, classes, interfaces.

@NgModule : it is type of normal class with @NgModule

It is collection of more than one components, service, pipe etc.

**declaration :** This property hold the details about all components.

**imports** : import pre-defined or user-defined modules.

BrowserModule which help to display the output in browser area or render the data in browser area.

**provider** : This properties help to provide angular service class details.

bootstrap : [AppComponent]

**main.ts**

from main.ts file it load the main or parent module details.

**index.html**

app.module.ts (parent module)

component1 order.module.ts

component2 component1

component3 component2

component3

**ng new angular-data-binding**

routing : no

styling :css

ng serve –o : after compiled project successfully automatically it open the project in default browser.

syntax to create the component using angular cli

ng generate component componentName

or

ng g c componentName

**Data binding** : Data binding is use to share the variable or property from component to view and vice-versa.

Data binding provide the bridge between component to view (template).

2 types

1. One way data binding
   1. String interpolation : component -🡪 View

{{}}

{{variableName}}

{{5+2}} {{6-3}}

{{sayHell()}}

* 1. Property binding : component --🡪 view

[]

<input type=”text” value=”Raj”/> static value

<input type=”text” [value]=”name”/>dynamic value

<p [innerText]=”name”></p>

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

()

Angular using same JavaScript event. But remove pre-fix on and event name wrap with ().

**DOM Event Angular Event**

onClick (click)

onclick (dblclick)

ondblclick (ngSubmit)

onSubmit (change)

onChange (keyup)

html – js template – component (ts)

jQuery events

$(document).onReady(function(){

$(“#b1”).click(function(){

alert(“hi”);

})

})

index.html

<input type=”button” valule=”click here” onClick=”fun()”/>

1. Two way data binding
   1. We can achieve two way data binding using event binding and string interpolation.

{}-🡪component -🡪 view

()--🡪view ----🡪component

🡨------------🡪

* 1. We can achieve two data binding using ngModel attribute.

[(ngModel)]

View🡨-----------🡪Component

ngModel is a pre-defined attribute which help to achieve two way data binding. This attribute is a part of FormsModule. So we have to import this module in module.ts file.

Passing the value textfield, passwordfield etc value from template to component.

Template reference

<input type=”text” #name/>

ProfileDetails Page

FirstName

Lastname

Gender

Hobbies

City

Submit

ng new angular-directive

ng new angular-forms

09-07-2021

Type of directives

Directives is use to add the extra behaviour to DOM elements.

3 types

1. Component directive : it is a type of directive which help to create the user-defined tags.

Component directive : It is use to create the user-defined tags using selector property. Whenever we use selector as a tag in any html page. The code which written in external template or internal template with data binding( one way or two way).

@Component({

selector:”my-Tag”, <my-tag></my-tag>

templateUrl:”./mypage.html”

})

class MyComponent extends Component {

set of variable and functions.

}

1. Structure directive : using structure directive we can add or remove DOM elements.

\*ngIf

\*ngFor

1. Attribute directive : attribute directive use to apply styling or css effect.

ngStyle : style

ngClass class

Template (View ) to Component

Component to View

Angular forms

Using Angular forms we can pass the more than one value as a group in the form of json.

2 types of

**Template driven form**

Template or view --------🡪 Component

It is very easy.

More code on View side using HTML Concept.

**Model Driven or reactive forms**

Component <---------------------View

It is complex. Good for develop for complex forms.

Mode code on Component side.

ng g c tdf-login-page

ng g c mdf-login-page

**Template Driven Form**

Template Driven form we can create the reference of form.

<form #loginRef = “ngForm”>

</form>

ngForm is a pre-defined attribute which help to create the reference or object of form in template driven form concept.

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

**Model Driven Form**

According to model driven form

TextField, PasswordField, radiobutton, checkbox, etc are known as FormControl

FormGroup is a collection of more than one FormControl as well as another FormGroup.

LoginPage

TextField

PasswordField

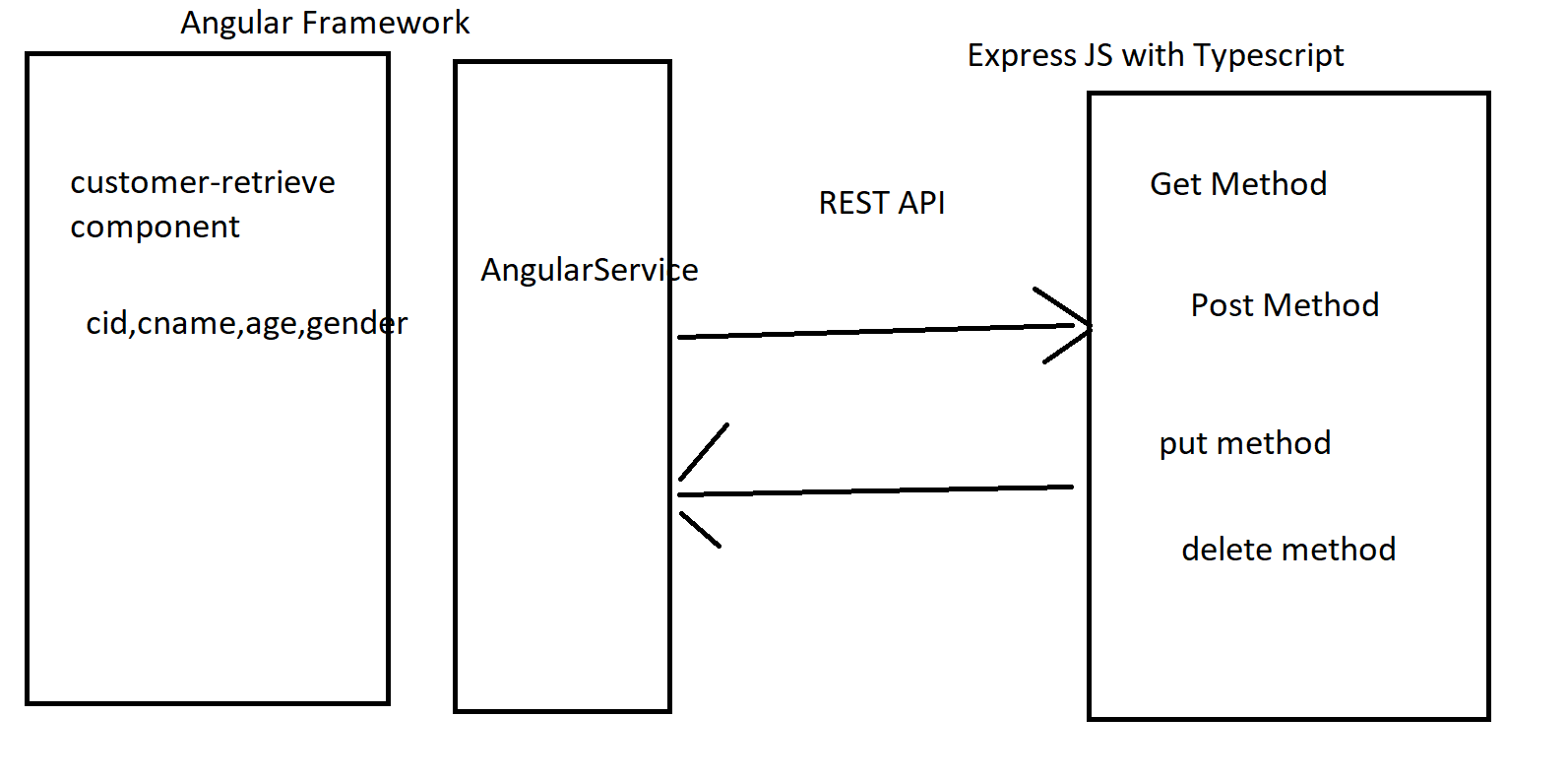
Submit Reset

LoginPage is known as FormGroup

TextField and PasswordField is as FormControl

formGroup and formControlName are pre-defined attribute part of ReactiveFormsModule.

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



ng new angular-service

tdf-login.html

html (view)-----------🡪Component

call service methods

mdf-login.html

html(view)-------------🡪component

call service methods

**service class**

checkLogin()

logic to check username and password

Service : Service class is use to make the separation of concern.

View Controller Model Layer

View (template)-----🡪Component -------🡪 Service

View (template--🡪Component

Angular Service

1. User-defined service
2. User-defined service object creation explicitly
3. User-defined service object created using DI (Dependency Injection concept).
4. Pre-defined service

HttpClient

**ng new angular-service**

ng g c first

ng g c second

IOC : Inversion of control : IOC is a concept. It is a type of design pattern. According to IOC place of creating any resource or objects explicitly allow to create and maintain by container. Object

We can describe or implement the object in JavaScript

Literal style

Function style

Class style

DI : Dependency Injection

DI of implementation of IOC.

let emp = new Employee(); we push the object in memory

let emp (try to pull the object from a container).

Constructor base DI

Setter base DI

Angular support only constructor base DI.

function Employee() {

}

let emp = new Employee();

User defined service using DI

First make normal class with decorator @Injectable

Register service class details in module level or component level using provider attribute.

Then each component achieve DI using constructor

If we register the user-defined service class on module level then it create only one object and many component can access. One object (single ton design pattern).

But if we register on component level then each component new memory get created.

Pre-defined Service :

Angular Provide pre-defined API

HttpClient :

HttpClient API contains http protocol methods like get, post, put and delete etc.

Which help to call REST API develop in any language

So first do the DI for HttpClient API.

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

.subscribe()

This function take 3 parameter as a callback function

next()

error()

()🡪 completed.

Fetch(“”) : This function return type is **promise**.

Then() and catch()

**HttpCliend** pre-defined API is part of HttpClientModule.

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

RxJS : Reactive JavaScript.

Reactive Programming using Java, Python, JavaScript etc

RxJS is a library for composing asynchronous data and events.

Reactive JS combines the Observable pattern with the iterator pattern and functional programming.

Observable : It represents the idea of invocation collection of future values or event or data.

One

Subscribe : represents the execution of Observable. It can be subscribe or unsubscribe (cancel).

Many

Observable vs Promise

Operator : are pure function that enables a function programming style dealing with collection of operations like map(), filter(), cancat etc.

Observable load the data one by one. Promise load whole data at time.

Observable load the data using subscribe method with three callback function next, error, ()-

Promise load the data using then and catch.

Observable can be cancellable using unsubscribe. Promise can’t cancel.

esm (embedded system module) This module will help to run the Observable application with import statement.

Npm install esm

To run the program we have to use the command as

node –r esm app.js

this.http.get(“URL”)

this.http.post(“URL”)

this.http.put(“URL”)

this.http.delete(“URL”)

.subscribe(next,error,())

RxJS Provide set of operator to filter the data.

nodemon : This is external Node JS Module which help to auto re-load or restart the application if we do any changes.

**npm install –g nodemon**

nodemon app.js

**backend technologies**

Mongo DB :

Collection: Projects

Standard 5 operation

getAppProject

getProjectById

storeProjectDetails

updateProjectInfo

deleteProjectInfo

FrontEnd

One operation = One component

One collection = One Service class.

In Angular Project

Component creation

**ng g c get-app-project-details.**

Creating service class through command prompot

**ng g s projects**

Two domain

Angular -🡪 4200

Express 🡪 9090

CORS : Cross Origin Resource Sharing.

For JavaScript Express JS

npm install cors

For TypesCript express JS

npm install @types/cors

To create the Model class using ng command

ng g class Projects --type=model

ng g interface Projects --type=model

create four components to store, retrieve by id, delete and update.

HttpClient all methods by default response is consider as json.

Rest API return types is

String : confirmation : success or failure

Return object : single object in json format or xml format.

Return objects : multiple or array object in json format or xml format.

Return : error.

HttpClient by default format consider as json.

Template Driven Form with Validation

Writing validation rules on view side.

With the help of html5 validation they provide few pre-defined classes to achieve the validation using template driven form.

**required**

**minlength**

**maxlength**

**pattern using regular expression**

In Angular few pre-defiend class are enable if we use required, minlength, maxlength, pattern.

ng-\*

ng-valid

ng-invalid

ng-touched

ng-untouched

ng-dirty

ng-pristine

templateRef.errors.required

templateRef.errors.minlength

templateRef.errors.maxlength

templateRef.errors.pattern

Model Driven Form

Writing the validation rules in component side.

\d 🡪 0 to 9

\D 🡪 non digit

\w🡪 word (letter and digits)

\S🡪 Non Space

\d{n}🡪 number of digits

\d{3)-🡪 any 3 digit numbers

a\d{3}🡪 start with a and 3 digit number

[abc]\d{3}🡪 a123, b345, c456

**Component Communication**

If any component contains properties or state variable of type number, string, boolean, array, complex object or array complex object.

The scope of those properties within that component or that component’s template.

class A {

n:number =10;

}

class B {

m:number = 20;

let obj = new A();

obj.n

}

**Sharing the data between the component**

1. Parent to Child -🡪 We have to take the help of @Input decorator.
2. Child to parent 🡪
   1. @Output decorator with EventEmmitter API
   2. @ViewChild decorator
3. Sibling
   1. sessionStorage and localStorage.
   2. Create common service class (shared service class) and register on module level. Then inside service class write the property of any type and provide setter and getter concept.
   3. Using RxJS concept we have share the data.

**Angular Routing**

Routing is use to navigate from one component to another component depending upon the path provided in routing file.

In routing we are replacing one component template by another component template using events with path.

Create 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 tags provider angular router which help to load the component’s template page contents depending upon the path providing the routing file.

It is like a place holder.

**Auth Guards**

**Angular provide lot pre-defined Auth guards. They are type of interfaces. Which provide set of life cycle methods which help to make the restriction on a path.**

**CanActive**

**CanDeactive**

**CanLoad**

**CanActiveChild**

**Etc**

**Agile**

**Testing : Testing is use to find the defect, error or bugs etc**

**Input a,b,sum**

**Process sum = a-b**

**Output write sum**

**2 types**

1. **Black box testing**

**Input Output**

1. **White box testing**

**Input Process Output**

**Unit testing : Unit testing is a kind of software testing method in which each individual and independent part of the source code is tested.**

**We write the code inside a function or methods or module or classes or packages**

**UI testing :**

**Backend testing using JavaScript :**

**BDD and TDD**

**Behaviour Driven Development : BDD is designed to test an application behaviour from the end user point. In this type of testing product manager, developer, tester and client are involved.**

**Test Driven Development : In this type of testing is a development features technique which focuses more on the implementation of a features of the project.**

**Node JS Testing**

**In Node JS contains pre-defined assert which help to do assumption of the result of the application. Which contains core functionality.**

**UI Testing Framework**

**Frontend JavaScript**

**JavaScript Jasmine and Karma**

**Jasmine**

**Karma : Angular framework**

**Mocha**

**Chai Node JS and Express JS Testing**

**superTest Http api testing**

**JEST : React JS**

**Etc**

**Testing framework**

**Test Suit : This construct is used to write the one or more than on test spec. To write the test suite every testing framework provided pre-defined function ie describe.**

**Jasmine, Mocha, JEST**

**Syntax**

**ES5 style**

**describe(“Operation Testing”,function() {**

**})**

**ES6 style**

**describe(“Operation Testing”,()=> {**

**})**

**Test spec : A test spec contains more than expectation result for the particular testing function. This construct test spec we have to use pre-defined function it.**

**describe(“Operation Testing”,()=> {**

**it(“Addition”,()=> {**

**more than one expect**

**})**

**it(“Sub”,()=> {**

**})**

**})**

**Expectation :**

**expect : expect function help use to check actual and expectation result from the function.**

**Mocha Testing**

**First create the new folder**

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

**npm init –y**

**or**

**npm init**

**npm install –g mocha**

**npm install –D mocha**

**Mocha is small testing library framework which provide test suite and test spec related function. It doesn’t provide expectation function.**

**Mocha take the help of other assertion library depending upon the requirements.**

**Chai**

**Should.js**

**Expect.js**

**Mocha with Chai**

**Chai assertion library**

**Please create new folder**

**Create package.json file**

**Using command as**

**Npm init –y**

**Npm install –D chai**

**Chai library we can use the function in different style**

1. **Assertion style: this provides the classical assertion dot notation like assert library of node js.**
2. **Expect style : In this style we can chain the together like a natural language assertion.**
3. **Should style: should style allow same as expect style chain of function. In this style each property or object with should property start your chain.**

**Create new folder for testing node js application**

**create package.json file**

**install mocha and chai dependencies**

**npm install –D mocha**

**npm install –D chai**

**Node JS Express JS Testing**

**Package json file using npm init**

**npm install express**

**npm install –D mocha**

**npm install –D chai**

**npm install –D chai-http**

**Jasmine and Karma**

**Jasmine is a open source testing framework which help to do the testing for JavaScript (Client side as well as server side scripting language).**

**UI Testing , HTML,CSS, Java using ES5.**

**Jasmine provided test suit, test spect and expectation functions.**

**Jasmine with client side JavaScript**

**Karma : Karma is a test runner for JavaScript library or framework. We can configure Karma with different browser to get the Testing result on browser.**

**Angular testing Jasmine and Karma.**

**Client Side JavaScript using Jasmine, Mocha, JEST with Karma Runner.**

**Jasmine with Server side JavaScript**

**Express JS Testing superTest library to do testing with Jasmine.**

**Angular Filter : Angular filter or pipe is use to filter the data or transform the data on template.**

**name:string =”Ravi”;**

**salary:number=15000;**

**dbo:Date = new Date();**

**Name is {{name}}**

**Name is {{name|uppercase}}**

**Name is {{name|lowercase}}**

**Salary is {{salary}}**

**Salary is {{salary | currency}}**

**Salary is {{salary | currency :’INR’}}**

**Date of joining is {{dob}}**

**Date of joining is {{dob|date}}**

**Date of joining is {{dob|date:’short’}}**

**Date of joining is {{dob|date:’long’}}**

**Date of joining is {{dob|date:’dd-MM-yyyy’}}**

**Angular Life cycle**

**Angular life cycle or hook function or methods.**

**Hook automatically get called.**

**Hook function divided into 3 parts.**

**Creates change destroy**

**Only once again and again only once**

**onInit2 doCheck 3 onDestory 8**

**AfterContentInit4 onChanges 1**

**AfterViewInit 5 AfterContentChecked 6**

**AfterViewChecked 7**

**Http is stateless protocol**

**Session Management or session track**

1. **Cookies : It is a small text file created by server when client send the request to server. This file store in client machine.**

**1st req**

**Client ----------🡪Req-------------🡪 Server**

**🡨----------Res-+Cookies(sessionId)----------------**

**1st Res**

**2nd Req + SessionId ------------🡪**

**JWT : Json Web Token**

**19-07-2021**

**Docker is a container.**

**Container is a part of server or engine responsible to execute or take care the life of the application.**

**VM**

**Base Window 16 RAM**

**GOS 4RAM CentOS**

**10 VM**

**16RAM**

**1Gb RAM**

**1**

**Docker is an advanced OS virtualization software platform that makes it easier to create, deploy and run the application in Docker container.**

**Virtualization and Containerization : Virtualization is an abstract version of physical machine. While containerization is the abstract version of an application.**

**Docker Container : Running the instance of Docker Image, container turn the actual application. A container includes an application and all of its dependencies.**

**Docker image : A Docker image contains everything you need to run your application. Image contains configuration details to run the Docker application.**

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

**Docker hub : Docker hub is like a github. Which is use to store the Docker images as well as we can publish and pull other images in local machine.**

**Docker Commands**

**docker -–version**

**docker images : This command to check all images present in our machine.**

**docker pull imageName : This command is to pull the Docker images**

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

**docker run imageName/imageId**

**docker images**

1. **Hello-world : running C Program**
2. **busybox :** BusyBox combines tiny versions of many common UNIX utilities into a single small executable.
3. **Docker info : This command provide us docker details.**
4. **alpine :**

**Create the image**

**By default Docker image file consider as Dockerfile (without extension)**

**docker pull akashkale/my-web:latest**

**to run the web application image we have to run the command as**

**docker run –d –p 9090:80 imageId**

**To view all running container**

**docker ps (process status)**

**To view all stopped and running container**

**docker ps –a**

**running node js program using docker**

**create angular project and then write coding.**

**After then build the project using command as**

**ng build**

**dist**

**Java + dist : deploy on server**

**Python**

**.net**

**Node JS +dist : deploy on server (normal server or actual production server).**

**Dist : folder going deploy on nginx server using Docker.**

**Front end : Angular**

**After created angular application**

**Using ng build command : build the projects.**

**Then copy and paste build file inside a node JS projects.**

**Backend : Node JS with TypeScript and express JS**

**In app.js or server.js or index.js (main file)**

**//Load the angular static files.**

app.use(express.static(process.cwd()))

// to load the first page of the application.

app.get("/",(req,res)=> {

    res.sendFile(\_\_dirname+"/index.html");

})

**Angular 4200**

**Angular : http:locahost:9999**

**Angular deploy on docker**

**8080**

**Angular image running on port number 8080**

**Express**

**99999**

**Express js deploy on docker then run express js application on 9999 port number.**

**Express JS image must be run on 9999 port number only.**

**To publish the image in Docker hub**

**First we have to create the tag (normally we use latest).**

**docker tag imageName:latest dockerHubId/imageName**

**after tag created**

**push the image using the command as**

**docker push dockerHubId/imageName**

**CI/CD : continuous Integration and Continuous Delivery**

**Code changes made by individual team members in their machine and merged together into working software, which is known as Integration phase.**

**Jenkin is open source software which help to do CI and CI.**

**CI is a development practise in which the developers are need to commit changes to the source code in a shared repository at regular interval period of time. Every commit made in the repository is build.**

**docker run –d -p 8080:8080 -p 50000:50000 -v jenkins\_home:/var/jenkins\_home jenkins/jenkins:lts-jdk11**

**docker logs containerId**

**admin password**

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

**username : admin**

**password :**

**Then it ask to install the plugin**

**It pull the image as well as run the image**

**docker run -p 8080:8080 -p 50000:50000 jenkins/jenkins:lts-jdk11**

**it require initial password**

**This command is use to find the initial password**

**docker logs containerId**

**username : admin**

**password : random generated password**

**it ask for the initial plugin . please install those plugins.**

**Now you can create the account.**

**UserName :**

**Password :**

**We can configure Jenkin with GIT**

**If any person commit and push the data in git in user-defined branch or main/master.**

**Jenkin -🡪 CI and CD**