**Day 1**

**24-05-2021**

**Web Developer**

**MERN Stack :**

**Mongo DB/MySQL Express Module React JS Node JS**

**Phase 1 :**

**Agile, Git, HTML,CSS, JavaScript**

**Project**

**Simply blogging**

**Phase 2**

**ES6 Features and React JS with Redux.**

**Project, Node JS Module**

**Building a To-Do App**

**Phase 3**

**Node JS Modules file handling, http module,**

**Express module (REST API),**

**Mongo DB,**

**Connecting Mongo DB data through Node JS**

**MongoDB and Mongoose**

**Socket programming**

**Project**

**Chatbox**

**Phase 4**

**Testing modules**

**Docker**

**Jenkin**

**AWS Overview ( ES3, RDS, S3 etc).**

**SVN : Java**

**Client1 or dev1 or Local Repository**

**Client2 or dev2 or Local RepositoryServer**

**Repository**

**Client3 or dev3 or Local Repository**

**GIT : Git is known as Sub Version Control System.**

**It is use to records the changes done in file or folder or application or projects.**

**Online shopping application**

**Git provide distributed sub version control system.**

**Login Module**

**Customer Module**

**Order Module**

**Manager module**

**Git commands**

**To make the folder as local repository as**

**git init**

**if you want to check the last command status we have run the command as**

**git status**

**If we want to move file from untrack phase to staging area. We have to run the command as**

**git add filename**

**if we want to move this file from staging area to local repository (folder).**

**git commit –m “Commit Message”;**

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

**git config –-global user.email “**[**abc@gmail.com**](mailto:abc@gmail.com)**”**

**git config -–global user.name “userName”**

**github : github is a part of Microsoft which provide remote repository for the git.**

**AWS**

**Azure**

**Google cloud**

**Oracle cloud**

**Etc**

**git add . ( all files and folder) adding the staging area.**

**Command to connect local repository to remote repository**

**git remote add origin URL**

**now to push the data to remote repository we have to use the command as**

**git push –u origin HEAD (u means upstream and HEAD last commit in that branch).**

**If we do any changes in local repository means created new file, updated existing file or deleted files**

**git add .**

**git commit –m “Message for that task”**

**git push –u origin HEAD**

**Steps :**

**git init**

**git status**

**git add .**

**git status**

**git commit –m “message for task”**

**git status**

**data store in local repository**

**git remote add origin URL**

**git push –u origin HEAD**

**next time or again and again whenever you do any changes in project or folder.**

**git add .**

**git commit –m “commit for task”**

**git push –u origin HEAD**

**Another way to make local folder as a local repository**

**git clone URL**

**git branch : A branch is simply light weighted movable pointer which hold more than one commit details.**

**When we create local repository we can see default branch ie master or main.**

**Syntax to create the branch**

**git branch branchname**

**To view branch names**

**git branch**

**To move use-defined branch**

**git checkout branchName**

**To delete the branch**

**git branch –D branchName**

**Command to create the branch and switch the branch**

**git branch branchname**

**git checkout branchName**

**Or**

**git checkout –b branchName**

**Manager**

**Manager create sample code and push in remove repository.**

**Ali Ajay**

**Git clone done by both developer**

**Login Application**

**BranchName BranchName**

**Ali\_Login Ajay\_Application**

**Git clone URL**

**: First time to download or clone remote repository in local machine.**

**And**

**git pull**

**: This command is use to updated new changes from remote repository to local existing repository.**

**git pull : we have run this command in default branch ie main or master.**

**git push : we have to push use-defined branch to remote repository**

**If we want to check all commit details**

**git log**

**Day 2**

**25-05-2021**

**HTML,CSS,JS ES5 and ES6. Bootstrap.**

**Web Application**

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

**http: hypertext transfer protocol : set of rules**

**s : secure**

**www : world wide web**

**google : domain**

**com : commercial**

**Uniform Resource locator**

**Req(https/http)---🡪**

**Client Server**

**🡨-----Res (http/https)**

**HTML/HTML5**

**CSS/CSS3**

**JS (JavaScript)**

**HTML/HTML5--🡪 Display the content on web page**

**Display simple message**

**Add the image**

**Hyper link**

**List**

**Table**

**EmployeeDetails**

**Id Name Salary**

**100 Raju 12000**

**101 Ramesh 14000**

**Attribute : Attribute is known as properties of tags. Attribute we have to write in opening in the form of key-value pairs. Value may be single quote or double quote or without quote.**

**<tagName name=”value”> </tagName>**

**Forms**

**HTML/HTML5 Form tag by default method consider as get.**

**If method is get the information send through URL in the form of query params. Like**

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

**In Get method data send through url ie through head part. And body is empty.**

**Post method**

**The data send through body part of request.**

**CSS/CSS3---🡪 Apply presentation logic or look and feel for that content.**

**CSS : Cascading Style Sheet :**

**CSS provide lot of pre-defined properties which help to apply good look and feel application.**

**Using CSS we can achieve separation concern. Means actual contents and formatting style of separate.**

**Types of CSS files**

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

**Inline CSS**

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

**</tagName>**

**Internal or Embedded CSS**

**Syntax**

**In between head tag we have to style stag**

**<style type=”text/css”>**

**selector {property : value;property:value}**

**</style>**

**Type of selector**

1. **Universal selector : \***
2. **Specific selector : tagname p, h1, to h6, div, span etc**
3. **Multi specific selector : tagname,tagName,tagName{property : value}**
4. **Class selector (local class selector ) :**

**tagName.className {property:value;property:value;}**

1. **Class selector ( global class selector )**

**.className {property:value;property:value;}**

1. **Id selector**

**#idName {property:value}**

1. **Child selector**

**parentNametag childTagName {property:value;}**

**Class Vs id**

**class : group of tags. So we can write more than one tag same class.**

**id : if we want to make unique ness between two tag when we have to use id.**

**<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>**

**Using the id in JS we can read, write or update html contents(DOM).**

**External CSS file**

**JavaScript --🡪 Action(Events) on content or programming on web page or contents.**

**UI Developer : IDE**

**Micro soft : VSCode**

**MEAN Stack and MERN Stack**

**JavaScript : ES5**

**JavaScript was object based interpreter scripting language.**

**Object based or prototype based style Vs object oriented**

**OOPs : Object Oriented Programming system**

**Object, class, Inheritance, Polymorphism, Encapsulation, Abstraction etc.**

**Like C++, Java, Python, C# etc**

**Interpreter Vs Compiler**

**Both are translator : converting one format to another format.**

**Interpreter : it check the code line by line and convert : JavaScript**

**Compiler : Convert whole code at time. C, Java**

**Scripting Vs Programming**

**In programming language it generate another file which help to run the program like byte code, exe file or object file.**

**But in scripting directly we get the output.**

**JavaScript tags syntax**

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

**</script> closing tag**

**This tag we have to write in between head tag or body tag of html we page.**

**We can write more than one tag in one html page.**

**If we want to display message through JavaScript we have to use pre-defined object ie document and write if function part of document object.**

**In JavaScript it is not mandatory every statement end with semicolon.**

**Variable and data types**

**To declare the variable in JavaScript we are using var keyword**

**var variableName**

**var a;**

**var n=10; // number type consider**

**var m=10.10; // number type consider**

**var fname=”Ajay Kumar”; //string type consider**

**var result = true; //Boolean type consider**

**var obj = new Date(); //object reference consider**

**document.write(a);// undefined (data type consider)**

**Operator :**

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

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

**Logical operator : &&, ||, !**

**Assignment operator : =**

**Increment and decrement : ++, --**

**Ternary operator : condition ? true : false**

**typeof function or operator :**

**Day 3**

**26-05-2021**

**If statements**

**Simple if**

**if(condition){**

**true block**

**}**

**if else**

**if(condition) {**

**}else {**

**}**

**Nested if : if within another if.**

**if(condition) {**

**if(condition) {**

**}else {**

**}**

**}else {**

**if(condition) {**

**}**

**}**

**If else if or if ladder**

**if(condition) {**

**}else if(condition) {**

**}else if(condition) {**

**}else {**

**}**

**switch :**

**syntax**

**switch(variableName) {**

**case label1: block1;**

**break;**

**case label2: block2;**

**break;**

**case label3: block3;**

**break;**

**default default bock**

**break;**

**}**

**switch, case, break and default are keywords.**

**Looping : it is use to execute the set of statement again and again till the condition becomes false.**

**While loop**

**Do while loop**

**For loop**

**Initialization start and end**

**Condition true**

**Coding**

**Increment / decrement**

**Normal Function**

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

**Syntax to write normal function**

**User-defined function**

**function functionName(parameterList) {**

**}**

**JavaScript function can return any type of values without return keywords or return type.**

1. **Function no passing parameter and no return type.**
2. **Function passing parameter but no return type.**
3. **Function passing parameter and return type.**
4. **Function no passing parameter but return the value.**

**Pre-defined global function**

1. **alert(“Message”): pop up message.**
2. **prompt(): This function help to take the value through keyboards.**
3. **parseInt() : it is use to convert string to integer.**
4. **parseFloat() : it is use to convert string to float.**
5. **eval() : it is use to convert string to number (with or without decimal).**
6. **confirm() : This function display pop message with 2 button ok and cancel. If you click on ok button it return true if click on cancel it return the false.**

**do {**

**alert 1: Add 2 Sub**

**prompt(): to receive choice**

**switch(){**

**case 1 take x and y value and display sum**

**case 2 take x and y value and display sub**

**}**

**Do you want to continue : confirm**

**}while()**

**alert(“thank you”)**

**events : event is a interaction between user and html tags or components or form tags.**

**Event provide bridge between html and JavaScript code.**

**Type of events**

**In JavaScript all events are start with on prefix followed by event name.**

**Type of events**

**onClick : button or p or div tags**

**onDblClick : button or p or div tag**

**onMouseOver : image tag**

**onMouseOut : image tag**

**onKeyUp : AJAX Google search engine**

**onKeyDown : chatting application**

**onFocus : enter in text field**

**onBlur : exit from text field**

**onChange : dropdown**

**onSubmit : Validation Form validation**

**onLoad : body tag**

**onUnload : body tag**

**Dropdown**

**<select name=”city” onChange=””>**

**<option value=”Bangalore”>Bangalore</option>**

**<option value=”Delhi”>Delhi</option>**

**<option value=”Mumbai”>Mumbai</option>**

**</select>**

**etc**

**Listener : Listeners are function which help to listen the generated events.**

**To generate the events we have to register the event on dom(Document Object Model) or HTML tags.**

**DOM : Document Object Model**

**index.html**

**In browser memory dom hierarchy will created.**

**<html>**

**<head>**

**<title>Message</title>**

**</head>**

**<body>**

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

**<script type=”text/JavaScript”>**

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

**document.write(“Welcome”)**

**}**

**</script>**

**</body>**

**</html>**

**DOM means HTML tags**

**HTML**

**HEAD BODY**

**TITLE P**

**TextNode –Message TextNode: Welcome to**

**If we want to write (add), remove, update dom elements or tags dynamically.**

**Read, Write and Update : dynamically**

**DOM API : Document Object Model (Application Programming interface).**

**Java, Asp.net, Python, C/C++, JavaScript they provide DOM API to read, write and update data in html dynamically.**

**DOM API using JavaScript**

**If we want to read text field value using JavaScript**

**1st way**

**document.formname.textfieldname.value**

**2nd way**

document.getElementsByName("user")[0].value;

**3rd**

**document.getElementById(“user”).value**

**Form Validation**

**Using JavaScript**

**Using HTML5 Features**

**OOPs using ES5**

**objects :**

**object : any real world entity**

**properties or state – have –fields/variables**

**Person**

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

**Bank**

**Animal**

**Car**

**Employee**

**Etc**

**In JavaScript object are divided into two types.**

**Pre-defined objects**

**JavaScript follow object hierarchy**

**Object -🡪 properties or state**

**behaviour**

**object 🡪 property or state**

**behaviour**

**object -🡪property**

**behaviour**

**object**

**BOM : Browser object Model**

**window is top most object in BOM Hierarchy**

**DOM : Document object Model**

**document is top most object in DOM Hierarchy**

**User-defined objects**

**Day 4**

**25-05-2021**

**BOM : Browser Object Model :**

**In BOM hierarchy window is a top most object.**

**window.alert(“Welcome to JavaScript”)**

**or**

**alert(“Welcome to JavaScript”);**

**window.prompt(“Enter the name”)**

**or**

**alert(“Enter the name”)**

**window.document.write(“Welcome to JavaScript”)**

**or**

**document.write(“Welcome to JavaScript”)**

**synchronous and asynchronous communication**

**Statement level**

**L1**

**L2**

**L3**

**L4**

**Function call**

**dis1();**

**dis2();**

**dis3();**

**dis4();**

**Promise()**

**1st Req**

**2nd Req**

**3rd Req**

**4th Req**

**Client Server**

**In JavaScript window object contains setTimeout(), setInterval() and clearInterval() function. Which help to achieve asynchronous operations.**

**document.write(“1st Statement”) syn**

**document.write(“2nd Statement”): asyn**

**document.write(“3rd Statement”) asyn**

**document.write(“4th Statement “) syn**

**CSS style property in DOM CSS property**

**color:red color=red**

**font-size:24px; fontColor=24px;**

**background-color:yellow backgroundColor=yellow**

**Creating User-defined object using ES5 style (function style)**

**Up to ES5 no class keyword to create the objects. To create the user defined object we are using function itself.**

**Object : any real world entity**

**properties**

**behaviour**

**object is a concept.**

**If we want to describe the object we have to take the help of function in ES5 JavaScript.**

**In function style object creation we can add dynamic property.**

**ES6 Features**

**ES : ECMA :European Computer Manufacture Association**

**ECMA Script : it is a concept.**

**ES5**

**ES6 and ES7 new**

**JavaScript is a one of the implementation of ES5 as well as ES6.**

**TypeScript is a super set of JavaScript which support all features of ES6. Where Javascript support few features of ES6 or partial support of ES6.**

**TypeScript is also one of the implementation of ES6 Features.**

**Adding external JavaScript file**

**From ES6 onwards we can declare variable using var, let and const keyword.**

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

**Using var we can declare global variable but using let we can declare local or block scope.**

**const : if we declare variable using const keyword we can’t change the value of that variable.**

**If we want to declare constant value using const keyword.**

**Normal function call may before or after function declaration.**

**But expression style calling must be after declaration.**

**IIFE : immediately invoked function expression**

**Syntax**

**1st 2nd**

**(functionBody)(functionCall)**

**This type of function we can call immediately and only once.**

**Callback function : passing the function it self or function name or function body to another function as a parameter is known as callback function.**

**Array with retrieve value using different ways**

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

**syntax to create the array**

**Literal style**

**let num1 =[100,200,300,400,500];**

**Object style**

**let num2 = new Array(100,200,300,400,500);**

**Day 5**

**30-05-2021**

**arrow function :**

**arrow function short cut syntax for the anonymous function with express style.**

**Syntax**

**Let/var functionName = ()=>document.write(“Arrow function”)**

**By default arrow function return result without return keyword.**

**If we want to write more than one line code in arrow function we have to use curly braces with or without return keyword.**

**Array methods**

**Push() : add the element at last**

**Unshift() add the element at beginning**

**Pop() : remove element from last**

**Shift() : remove element from beginning**

**Splice(): This method is use to add, remove and update the elements.**

**Array de structure concept.**

**Rest Operator and Spread Operator**

**Syntax Rest and Spread operator**

**…variableName**

**We can use with array**

**For Rest Operator …variablename on the left side of the de structure syntax.**

**For Spread operator …variable name on the right side of the array variable.**

**Spread operator**

1. **spread operator is use to create the clone of the array or duplicate array.**
2. **Spread operator is use to merge more than one array object.**
3. **Using spread operator we can merge two use-defined object property into one objects.**

**We can use with function**

**Creating user-defined object using ES6 style**

**ES6 style OOPs concept.**

**From ES6 onward we create to describe object we are using class keywords.**

**ES6 style**

**class className {**

**variableName = value;**

**variableName= value;**

**functionName() {**

**}**

**}**

**constructor**

**constructor is a type of special function which help to create the object.**

**Up ES5 function itself is behave like a constructor.**

**But from ES6 they provide constructor. If we want to write constructor from ES6 we have to create a function with name as constructor in lower case.**

**Constructor get called automatically when we create the objects.**

**But in ES6 we can write only one constructor means we can’t write more than one constructor.**

**Day 6**

**31-05-2021**

**DOM Operation Insert and Delete DOM Element dynamically.**

**Project Details**

**Name\_Organization\_MERN\_Stack\_Repotitory :Repository**

**Local Machine**

**Create one folder**

**Phase 1**

**Create a Phase1 Project**

**Phase 2**

**Phase 3**

**You have to push this code in git repository.**

**Web Service :**

**Req java(req)**

**Client SBI XML/JSON HSBC**

**Spring boot asp.net**

**Res python**

**Node js Non java res**

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

**SOAP Base only XML format**

**REST API Web Service xml and non xml ie json**

**Any other format.**

**JSON : JavaScript Object Notation**

**JSON is use to store the data in the form of key value pairs. Where key must in string format and value may be number, Boolean, string, array, complex object.**

**Syntax**

**{“key”:value,”key”:value,”key”:value}**

**In View side we have understand how to convert string to json and json to string.**

**Promise : Promise is a pre-defined object provided by JavaScript which help to handle asynchronous action’s eventually may be success values or failure values.**

**User-defined promise**

**Let pr = new Promise((res,rej)=> {**

**res(“successfully done”)**

**})**

**//promise handle asynchronous event data if promise resolved then callback function get called if promise rejected catch callback function call.**

**pr.then().catch()**

**JavaScript provide pre-defined method/function fetch() which help to consume/produce REST API from backend technologies develop in any language Java(Spring boot), Asp.net, Python, or Node JS.**

**Fetch() function return type of promise objects.**

**Rest API**

**Representational State Transfer Application Programming interface.**

**Backend technologies like Java, .net, python or Node JS expose their data (services) in the form of JSON using Rest API.**

**URL if they given data in the form of JSON (XML) They are REST api develop in any technologies.**

**HTML5 : Storage API**

**sessionStorage and localStorage**

**If we want to share the data between more than one JavaScript file may be ES5 or ES6 we can take the help of Storage API.**

**store.js**

**var a=10;**

**retrieve.js**

**document.write(a)**

**If we want to share the data between two js file then we can take help of sessionStorage or localStorage**

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

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

**key must be string**

**sessionStorage.getItem(“key”);**

**localStorage.getItem(“key”);**

**using key we can get the value from session scope.**

**This value is available till application close. Once application close the value form session store destroy.**

**Some time if you want to remove the value from session storage we can call**

**sessionStorage.removeItem(“key”);**

**If we store the value in localStorage it store the secondary memory. After close the after still we can get the value we open the application once again.**

**localStorage.removeItem(“key”);**