**Day 1 : 03/01/2022**

**Phase 3**

**Create scalable and dynamic web site**

**Scripting language**

**Client side and server side scripting language**

**What is node js**

**Node JS modules**

**Types of module**

**Core module**

**User-defined module**

**External module**

**Fs module**

**http module**

**url module**

**express module**

**rest full web service using express module**

**mongo db database**

**basic and adv mongo db database query**

**connecting mongo db database using mongodb and mongoose module**

**express mvc**

**MEAN Stack**

**Front end angular**

**Backend express and mongo db database**

**Socket.io programming**

**Front end side**

**Html /html 5 web page**

**Css /css 3 formatting style or presentation logic**

**Bootstrap**

**JavaScript : Validation on client and dynamic web page**

**Or action on web page or event on web page.**

**jQuery**

**Backend technologies**

**Java (JEE) : Servlet/JSP or Spring framework or spring boot**

**Asp.net**

**Php**

**Python**

**Node JS**

**Before Node JS lot of library and framework using JavaScript**

**jQuery**

**Angular JS**

**Angular Framework**

**React JS**

**Vue JS**

**Backbone js**

**Ext js**

**Coffee js**

**Etc**

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

**Before NODE JS we were running JavaScript program using browser. Browser contains run time environment for Java Script (client side scripting language).**

**Node Js contains lot of pre-defined module that may be core module or external module which help to do operation using JavaScript**

1. **File handling program**
2. **Creating dynamic web application**
3. **Creating rest full web service**
4. **Connection database may be my sql or mongo db**

**Etc**

**REPL terminal: read eval print loop**

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

**Means js doesn’t provide BOM and DOM**

**Browser object model**

**Document object model**

**BOM and DOM available in client side scripting language not server side scripting language.**

**Node js provided pre-defined object ie global object**

**console.**

**Node JS FS Module**

**Fs means file system**

**Node Js provide lot of modules**

**Modules : modules is a combination of one or more than one files. Which help to do re-usability.**

**We connection one module to another module using require or import and export keywords.**

**In node js modules are divided into 3 types**

1. **Core modules (by default available with node js software).**
2. **User-defined module**
3. **External module**

**FS module : It is a type of core module which help to do file handling program synchronously or asynchronously.**

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

**read and write operation using asynchronously as well as asynchronously.**

**We store and retrieve data in string format.**

**We want to store the JS objet**

**We have to convert into json format.**

**In JavaScript we can create the object using**

1. **Function style**
2. **Class style**
3. **Literal style**

**JSON.stringify() method code help to convert JS object to string format.**

**Day 2 : 04/01/2022**

**Taking the value through keyboards in Node JS**

**readline : readline is a type of core module which help to take the value through keyboards in node js. Readline method are asynchronous methods.**

**Node js provided pre-defined global object is process**

**readline-sync it is external module which help to take the value through keyboards synchronously.**

**Syntax to install external module**

**Npm install –g moduleName globally**

**Or**

**Npm install modulename locally**

**http : hyper text transfer protocol.**

**Node js provide one of the pre-defined core module ie http which help to create the server side program using JavaScript.**

**Java (Spring framework or spring boot)**

**.net (asp .net)**

**Php**

**Python with django framework.**

**To run above application we require server.**

**Server is a like a engine which contains container which is responsible to execute java or asp.net or php or python.**

**Tomcat**

**WebLogic**

**IIS server**

**Apache**

**Mamp**

**Etc**

**XAMPP**

**All above server is thread base server.**

**Program : set of instruction to perform specific task.**

**Processor :processor is responsible to execute the code.**

**Process : time taken to execute the code or program in execution.**

**Thread : it is small execution of a code with in process.**

**Thread also known as light weighted process.**

**Multi tasking**

**Process base**

**Thread base**

**Multi tasking using thread base is faster than process base.**

**JavaScript is a not a multi threaded base scripting language it is a single thread scripting language.**

**By default all server are thread base server.**

**class Booking {**

**avl =1;**

**}**

**10 client send the request at the same time to this application.**

**1st client**

**2nd client**

**3rd client**

**10th client**

**If sever side technologies is multi threading for each client rather than creating separate memory it will create one memory and each client consider as a one thread.**

**thread can be lock or block.**

**Server can response concurrently 1000 or 10000 0r 100000 or 1000000 client at the same time.**

**Server has 100 client ( 100 thread created)**

**101**

**Node JS providing one of the great features ie**

**Event loop architecture**

**Create the server side program using http module with node js**

**url module : node js provided one of the pre-define core module ie url module which help to provide URL details.**

**When we use urlRef.parse(url) The query property consider as a string.**

**urlRef.parse(url,true); The query property consider as a reference.**

**http module with route concept**

**Day 3 :**

**05/01/2022**

**Node JS provided lot of pre-defined external modules**

**Which help to create the web page.**

**Express module : Express is a type of third party module which internally use http module and wrap this module and provide extra functionality to develop web application using node js.**

**While creating node js application we have to depends upon lot of external module.**

**Those external module we can install locally or globally.**

**If we install the module locally in the current folder node\_module will download and it contains all necessary files which help to develop the application.**

**package.json it is type of json file which hold configuration details about the node js projects.**

**Command to create the package.json file**

**npm init**

**when we run npm install command. Node js search package.json file in current directory and it download all dependencies in current machine available in packge.json file.**

**Node js provided pre-defined property ie**

**\_\_dirname : this is use to give the current directory path.**

**To enable the request body data in express js we have to use middleware. Middleware means intermediate between client and server.**

**Middleware module name is body-parser.**

**Old version of express js we have to download the body-parser module separately but new version of expression js this module available in express js module.**

**npm install body-parser**

**Day 4 :**

**06/01/2022**

**View is normal html page.**

**If we want to make dynamic view.**

**Express JS introduce View engine.**

**Those view engine are dynamic. So we can do any programming on that view.**

**If we use pug or jade or other view engine those view tightly coupled with express js application.**

**Java technologies**

**Amazon shopping application**

**Payment gateway**

**Credit card python**

**Debit card asp.net**

**Net banking php**

**Google pay java**

**Phone pay python**

**Paytm node express js**

**XML :eXtensible markup language**

**XML is heavy**

**JSON JavaScript Object Notation.**

**JSON is light weighted.**

**Web Service : Giving the service for web application when both the application running using different technologies and in same OS or different os.**

1. **SOAP base web service : Simple Object Access Protocol. In SOAP web service we can share the data only in the form of XML.**
2. **RESTfull Web Service : Representational state transfer protocol. Using Restfull web service we can share the data in any format ie xml or json or plain text or html or any other format.**

**Using Rest Full web service exposing our resources as a web service. So if we expose express js as a rest full web service. Any application can all our application**

**Like Java, Python, Asp.net, Angular , React js or any rest client application.**

**If we make express js as a rest full web service**

**Those service we can all using Angular.**

**So View – Frontend – Angular**

**Backend – express js work independently.**

**REST API (Application Programming interface).**

**According to REST API we have to work on http methods**

**Resource : it can be any entity like**

**Employee, Product, Order, Manager, Customer, Bank, Account etc.**

**Get method : get the resources**

**Get Employee details, get product details.**

**Get Employee details using specific property like id, age , salary desg etc.**

**Select query**

**Select \* from employee**

**Select \* from employee where age > 21;**

**Select \* from employee where name like ‘Ravi’;**

**Post method : create the resource**

**Storing employee details**

**Storing customer information**

**Insert query**

**Put method : update the resource**

**We update all property of resource using another property.**

**Employee id,name,age**

**We update name and age using id property**

**Update query**

**patch method : update the resource**

**we update partial property of resource using another property**

**Employee id,name,age**

**We update age using id property but not name**

**Update query**

**Delete method : delete resource**

**Delete employee details using id**

**Delete query**

**Create the folder Express JS REST API**

**Create the package.json file using npm init command**

**npm install express**

**get REST API method we can test through browser.**

**get method**

1. **Get data in string format**
2. **Get data in json format**
3. **Get employee details in json format**
4. **Get all employee details in json format**

**While calling rest api by any technologies like java or python or angular if they want to pass the value to rest api so we can use two technique.**

1. **Query param**
2. **Sending single value**

**URL?key=value**

1. **Sending multiple value**

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

**Path param**

1. **Sending single value**

**URL/v1**

**url/Ravi**

1. **Sending multiple value**

**URL/v1/v2/v3**

**url/100/Ramesh/21**

**if client application is normal html page**

**then can use query param. Because html form with get method internally use query param concept**

**If client is angular or react or other technologies then we can use path params.**

**Post method :**

**Post method is use to store the data or resources.**

**We can’t test post, put, patch and delete methods through URL.**

**To receive the data as json format from request body we have to use middleware.**

**Crud operation on entity product or employee or customer**

**Product id, name, price, images**

**Create the folder Product CRUD Operation folder**

**Create two sub folder**

**Backend**

**Create package.json file using command as npm init**

**And install**

**npm install express**

**5 Rest API Created**

1. **Get all product details**
2. **Get product info using pid**
3. **Store product information**
4. **Update product price using pid**
5. **Delete product info using pid**

**Frontend**

**Angular : using angular framework we will call these REST API.**

**In front end folder create angular project using command as**

**ng new angular-product-crud**

**routing 🡪 yes**

**styling 🡪 css**

**Once project created please run this command to run the angular project.**

**ng serve –o**

**we are going to call backend service ie product service which contains 5 REST API.**

**In Angular first we have to create the product component, product service and product model class.**

**ng g c product this command is use to create component**

**ng g s product this command is use to create service**

**ng g class product this command is use to create the model**

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

**We have to import this module in app.module.ts file**

**DOM Event**

**<input type=”button” value=”click”**

**onClick=”fun1()”/>**

**fun1 function is a part of javascript file**

**Angular Event**

**<input type=”button” value=”click”**

**(click)=”fun2()”/>**

**Fun2 function is a part of component ie typescript file**

**Event binding is a type of one way data binding ie template ----🡪 Component**

**()**

**Property binding is a type of one way data binding ie Component ---🡪 Template**

**[]**

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

**We are running two server application**

**One is backend server using express js with port number 9090.**

**Another one is front server using angular framework with port number 4200**

**CORS : Cross Origin Resource Sharing**

**In Backend technologies we have to enable cors policy.**

**Node js provided external module ie cors.**

**We have to install this module and use as middleware to enable the cors policy.**

**11-01-2021**

**To store the information permanently we are using file system.**

**Node JS fs module to store and retrieve the data.**

**Limitation of file base system**

1. **File system is not consistence. Format of the file**

**Id/name/salary**

**Id Name Salary**

**Id\_Name\_Salary**

1. **Data redundancy means duplicate records we can store.**
2. **Security. File security read or write mode.**

**Data : data is known as raw fact.**

**Information : processed data or meaningful data.**

**Database We are storing the data in table format. Schema. It is a software which help to store the data in table format.**

**DBMS Database management system**

**RDBMS Relational database management system.**

**Codd’s rules 12 rules.**

**Database : My SQL or oracle or POSTGres or db2 or any other database.**

**Excel sheet or open excel or excel software.**

**DBMS**

**Srno Name Age**

**100 Ravi 21**

**101 Ramesh 22**

**100 Ajay 23**

**StudentTrainerDetails**

**TId TName Tech Sid SName Age**

**1 Raj Java 100 Reeta 21**

**1 Raj Java 101 Meeta 22**

**1 Raj Java 102 Leeta 23**

**1 Raj Java 103 Meeta 24**

**Trainer**

**PK**

**TI TName Tech**

**1 Raj Java**

**Student**

**PK FK**

**Sid SName Age TSId**

**100 Reeta 21 1**

**101 Meeta 22 1**

**102 Leeta 23 1**

**No SQL Database**

**Not only SQL**

**Mongo DB database : Mongo DB is a type of No SQL open source document base schema less database.**

**Customer in RDBMS**

**Cid Name Age city PhNumber**

**100 Ravi 21 null null**

**101 Ramesh 22 null null**

**102 Ajay 23 Bangalore null**

**103 Mahesh 24 null 1234**

**In Express JS we have to convert JS object into table format.**

**We if retrieve query or records from RDBMS we have to convert all query in JS or JSON Format in Express js or node js application.**

**After installation of to run the server we have to**

**Use the command as**

**mongod**

**then you have to run the command as**

**mongo**

**To run the mongo shell**

**Once you connected database using mongo command**

**To check all database present in mongo db database we have to command is**

**show dbs;**

**or**

**show databases;**

**command to create the database**

**use meanbatch2;**

**This command is use to create the database if database not present else it switch to existing database.**

**In mongo DB table is known as collection.**

**To check collection or table present in database we have to use the command as**

**Show collections;**

**Or**

**Show tables;**

**Command to create the collection**

**db.createCollection("Sample");**

**in Mongo DB record is known as document.**

**So collection is use to store more than one document.**

**db.collectionName.insert({key1:value1,key2:value})**

**db.Sample.insert({name:"Ravi"});**

**To view the documents from a collection.**

**db.Sample.find();**

**mongo db internally create \_id as a pre-defined property with unique value. If you want to pass user-defined value you can insert it. But you can’t change field name or property name.**

**in mongo DB we can insert the document in collection without creating the collection. If collection already present it will insert in existing collection else it will create.**

**If we want to retrieve specific document from a collection using index position.**

**db.Emp.find()[1];**

**it will display complete document from 1 index position**

**db.Emp.find()[2].age;**

**it will display 2 index position name property value.**

**To retrieve one field or more than one fields value from a collection.**

**db.collectionName.find({condition},{key1:1})**

**db.Emp.find({},{name:1}); it display name and \_id fields**

**db.Emp.find({},{name:1,\_id:0}); it display only name field**

**db.Emp.find({},{name:1,age:1,\_id:0}); it display name and age field**

**below query is use to retrieve more than field with index position**

**db.Emp.find({},{name:1,age:1,\_id:0})[1];**

**condition to filed the documents like a where clause in RDBMS**

**db.collectionName.find({condition});**

**operators**

**db.Emp.find({age:{$gt:25}});**

**db.Emp.find({age:{$gte:25}});**

**db.Emp.find({age:{$lt:25}});**

**db.Emp.find({age:{$lte:25}});**

**db.Emp.find({age:{$eq:25}});**

**db.Emp.find({age:{$ne:25}});**

**to check more than one condition with and / or operator**

**db.Emp.find({$and:[{name:"Ajay"},{age:21}]});**

**db.Emp.find({$or:[{name:"Ajay"},{age:22}]});**

**sort the document using any field in ascending or descending order.**

**db.Emp.find().sort({age:1}) ascending order**

**db.Emp.find().sort({age:-1}) descending order**

**update document**

**db.collectionName.update({condition},{$set:{key:value}}**

**)**

**db.Emp.update({city:"Bangalore"},{$set:{city:"Mysore"}});**

**it update only one document**

**db.Emp.updateMany({city:"Bangalore"},{$set:{city:"Mysore"}});**

**This will update many document.**

**Remove document**

**db.Sample.remove({}); This query remove all document from a collection without conditions.**

**db.Emp.remove({\_id:1}); This query remove document with conditions.**

**Storing array value in collection**

**db.Student.insert({\_id:1,sname:"Ravi",age:18,sub:["Math","GK","English"]});**

**db.Student.insert({\_id:2,sname:"Ramesh",age:17,sub:["Math","Bio","Computer"]});**

**db.Student.insert({\_id:3,sname:"Ajay",age:19,sub:["Phy","Chem","Maths"]});**

**retrieve document from a collection with array value.**

**db.Student.find({sub:"Math"})**

**Monogo DB Relationship**

**One to one one mean pk and many mean fk**

**One to many**

**Many to one**

**Many to many**

**In mongo db we can achieve relationship using**

**Two ways**

**Embedded style relationship only one collection**

**Employee**

**Id, name, salary**

**Address**

**City, state**

**One to one (one employee has only one address)**

**db.Employee.insert({\_id:1,name:"Ramesh",salary:25000,address:{city:"Bangalore",state:"Kar"}});**

**one to many (one employee can have more than one address)**

**db.Employee.insert({\_id:3,name:"Ajay",salary:22000,address:{city:"Bangalore",state:"Kar"},projects:[{pid:1111,tech:"Java"},{pid:2222:"Python"}]});**

**one to many (one employee can work in many projects)**

**db.Employee.insert({\_id:3,name:"Ajay",salary:22000,address:{city:"Bangalore",state:"Kar"},projects:[{pid:1111,tech:"Java"},{pid:2222,tech:"Python"}]});**

**condition for complex property**

**db.Employee.find({"address.city":"Mumbai"}).pretty();**

**Linking style relationship we can use more than one collection**

**Trainer**

**\_id TName tech**

**1 Raj Java**

**2 Ravi Python**

**3 Ramesh Angular**

**db.Trainer.insert({\_id:1,tname:"Raj",tech:"Java"});**

**db.Trainer.insert({\_id:2,tname:"Ravi",tech:"Python"});**

**db.Trainer.insert({\_id:3,tname:"Ramesh",tech:"Angular"});**

**Student1**

**\_id SName Age tsid like FK not FK**

**100 Seeta 21 1**

**101 Reeta 22 1**

**102 Meeta 23 2**

**103 Keeta 24 2**

**104 Leeta 25 3,1**

**db.Student1.insert({\_id:100,sname:"Seeta",age:21,tsid:db.Trainer.find()[0].\_id})**

**db.Student1.insert({\_id:101,sname:"Reeta",age:22,tsid:db.Trainer.find()[0].\_id})**

**db.Student1.insert({\_id:102,sname:"Meeta",age:23,tsid:db.Trainer.find()[1].\_id})**

**db.Student1.insert({\_id:103,sname:"Keeta",age:23,tsid:db.Trainer.find()[1].\_id})**

**db.Student1.insert({\_id:104,sname:"Leeta",age:23,tsid:[db.Trainer.find()[1].\_id,db.Trainer.find()[0].\_id]});**

**db.Student1.insert({\_id:105,sname:"Yeeta",age:23,tsid:[db.Trainer.find()[1].\_id,db.Trainer.find()[2].\_id]})**

**Student2**

**\_id SName Age trainers**

**100 Seeta 21 1,Raj,Java**

**101 Reeta 22 1,Raj,Java**

**102 Meeta 23 2,Ravi,Python**

**103 Keeta 24 2,Ravi,Python**

**104 Leeta 25 [{3,Ramesh,Angular}**

**,{1,Raj,Java}]**

**db.Student2.insert({\_id:100,sname:"Seeta",age:21,trainers:db.Trainer.find()[0]});**

**db.Student2.insert({\_id:101,sname:"Reeta",age:22,trainers:db.Trainer.find()[0]});**

**db.Student2.insert({\_id:102,sname:"Meeta",age:23,trainers:db.Trainer.find()[1]});**

**db.Student2.insert({\_id:103,sname:"Keeta",age:24,trainers:db.Trainer.find()[1]});**

**db.Student2.insert({\_id:104,sname:"Leeta",age:25,trainers:[db.Trainer.find()[2],db.Trainer.find()[1]]});**

**db.Student2.insert({\_id:105,sname:"Yeeta",age:26,trainers:[db.Trainer.find()[1],db.Trainer.find()[2]]});**

**if we want to retrieve more than one fields from more than one collection then we have to use aggregate function with lookup operator**

**db.Student1.aggregate([{$lookup:{from:"Trainer",localField:"tsid",foreignField:"\_id",as :"TrainerInfo"}}]).pretty();**

**13-01-2022**

**Aggregate operator :**

**Mongo DB using aggregate operator groups multiple document and then perform some aggregate operation on those documents and it return the result.**

**Making the group by deptId**

**db.Employees.aggregate([{$group:{\_id:"$deptId"}}]);**

**Making the group by city**

**db.Employees.aggregate([{$group:{\_id:"$city"}}]);**

**db.Employees.aggregate([{$group:{\_id:"$deptId",totalSalary:{$sum:"$salary"}}}]);**

**db.Employees.aggregate([{$group:{\_id:"$deptId",maxSalary:{$max:"$salary"}}}]);**

**db.Employees.aggregate([{$group:{\_id:"$deptId",minSalary:{$min:"$salary"}}}]);**

**db.Employees.aggregate([{$group:{\_id:"$deptId",AvgSalary:{$avg:"$salary"}}}]);**

**connection mongo db database using node js application**

1. **Using mongo db module**
2. **Using mongoose module**

**Using mongo db module connecting mongo db database.**

**Mongo db is a external module we have to install it using command as**

**npm install mongodb**

**so first we have to create package.json file using command as npm init**

**mongoose :mongoose is a external module which internally follow ODM (Object Data Modeling) concept.**

**Using mongoose we an provide schema for the collection.**

**ORM**

**Mongoose internally use mongodb module only to connect the database. Mongoose module wrap mongo db module and provide extra features to develop enterprise application.**

**Mongo db module use native api driver to connect the database.**

**Create the folder as mongoose db module**

**Then create the package.json file using npm init command.**

**npm install mongoose**

**in mongoose we have to create the Schema. Schema help to provide the Structure for the collection.**

**Once schema is ready now we have to define the model.**

**Model take the help of schema and provide the collection name and using the model reference we can do all operation on collection in database.**

**By default mongoose collection name consider as plural names and in lower case.**