***Code step by steps***

***Daily Tuition***

1. *What is nodejs introduction*
2. *Basic point before start with code*
3. *Install node, npm*
4. *Make first program*
5. *Javascript fundamental in node js*
6. *Code modules*
7. *Create server*
8. *All about package.json file*
9. *A challage and 2 important interviews question*
10. *All about nodemon package*
11. *Simple and basic apis*
12. *Getting input from command line*
13. *Display file list from folder*
14. *Asynchronous programming language*
15. *Asynchronous programming language*
16. *Handle asynchronous data*
17. *How node js works*
18. *Express js introduction*
19. *Routing , params, request , response*
20. *Render html and json*
21. *Make html page*
22. *Remove extension from url and 404 pages*
23. *EJS template engine*
24. *Dynamic page with ejs*
25. *Express js middleware*
26. *Route level middleware express js*
27. *Install mongodb*
28. *Mongo db basic -1*
29. *Mongodb crud operation*
30. *Connect mongodb with nodejs*
31. *Read data from mongodb*
32. *Insert data in mongodb*
33. *Update data in mongodb*
34. *Delete records in mongodb*
35. *Basic get api with mongodb*
36. *POST api , insert data in mongodb*
37. *PUT api, update data in mongodb*
38. *DELETE API, delete data from mongoDB*
39. *Mongoose with node*
40. *Crud with mongoose*
41. *Post api with mongoose, how to make post api in node js*
42. *Get, delete, put api, method with mongoose*
43. *Search api with multiple fields*
44. *Upload file*
45. *Os module*
46. *Events and event emitter*
47. *Repl – read – eval-print-loop*
48. *Connect with mysql*
49. *Get api with mysql*
50. *Post api with mysql*
51. *Put api with mysql*
52. *Delete api with mysql*

*-php vs node js*

*-javascript vs php*

*-react and node js project*

*-node js complete tutorial*

*-node js interviews*

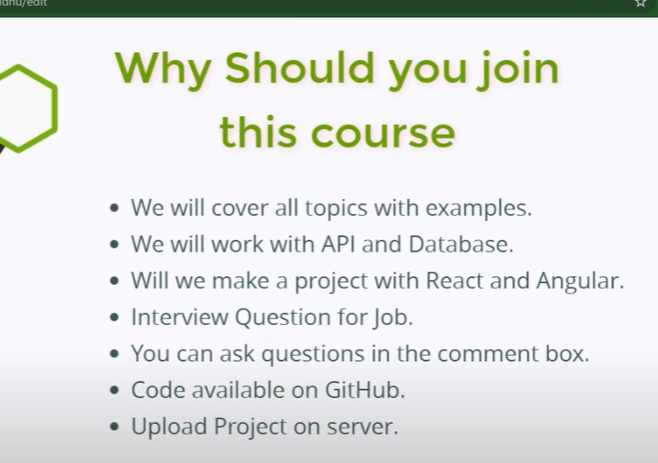
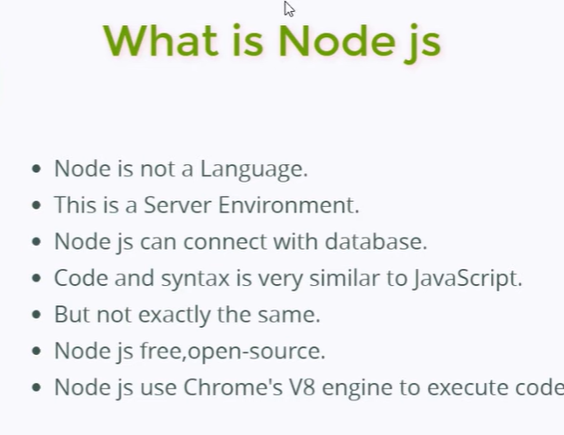
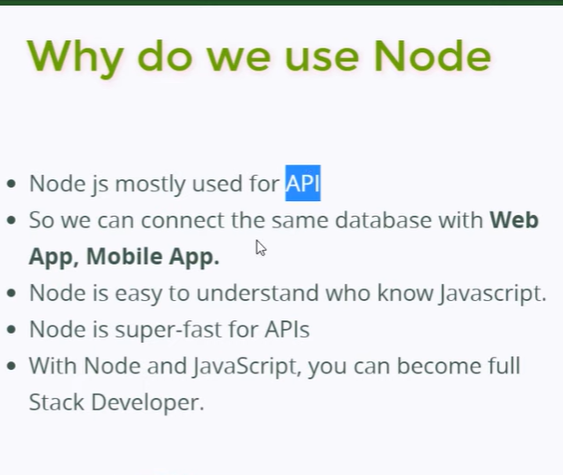
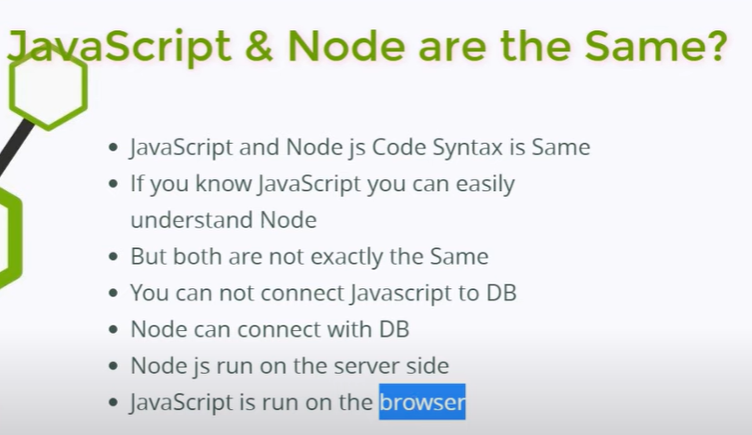
*-node js and jwt token authentication  
  
  
Github – code step by step – go through every github branch*[*Branches · anil-sidhu/node-js (github.com)*](https://github.com/anil-sidhu/node-js/branches/stale?page=1)

*To practice javascript – compiler*[*Online JavaScript Compiler (Editor) - Programiz*](https://www.programiz.com/javascript/online-compiler/)

[*Branches · anil-sidhu/node-js · GitHub*](https://github.com/anil-sidhu/node-js/branches/stale)

[*GitHub - anil-sidhu/node-js*](https://github.com/anil-sidhu/node-js)

[*Branches · anil-sidhu/node-js · GitHub*](https://github.com/anil-sidhu/node-js/branches/stale)below github node js course same as code step by step tutorial *https://github.com/Subham-Maity/node-js-full-stack-tutorial*[*Node JS in Hindi #1 what is Node | Introduction (youtube.com)*](https://www.youtube.com/watch?v=zaLfOjNEOaQ&list=PL8p2I9GklV456iofeMKReMTvWLr7Ki9At)

*Node JS in Hindi #1 what is Node | Introduction  
  
  
  
  
  
  
  
Node JS in Hindi #2 Basic Point Before Start with Code*

*Node JS in Hindi #3 Install node , NPM  
node module*

*package. Json*

*cmd - npm -v*

*cmd - node -v*

*cmd - console.log(“hello world”);  
  
Node JS in Hindi #4 Make First Program  
cmd – npm -v*

*cmd – node -v*

*node 🡪enter  
console.log(“hello”)*

*index.js*

*var x = 10;*

*var y = 20;*

*console.log(x+y);*

*run node index.js*

*console.log*

*console.error*

*console.warn*

*Node JS in Hindi #5 JavaScript Fundamentals in Node*

*Node JS import / export*

*1.pattern*

*import or export single variable and function*

[*https://youtu.be/F\_GmxjP80so?list=PLC3y8-rFHvwh8shCMHFA5kWxD9PaPwxaY*](https://youtu.be/F_GmxjP80so?list=PLC3y8-rFHvwh8shCMHFA5kWxD9PaPwxaY)

*math.js*

*const add = (a, b) =>{*

*return a+b;*

*}*

*//import or export single variable or function*

*module.exports = add;*

*index.js*

*const add = require("./math");*

*console.log(add(2,3));*

*2. pattern*

*Instead of having module. Exports , its on separate line, we directly assign arrow function to module. Export*

*The import will remain the same   
math.js*

*module.exports = (a, b) =>{*

*return a+b;*

*}*

*Index.js*

*const add = require("./math");*

*console.log(add(2,3));*

*3. Patterna*

*export , more than one variable or function*

*math.js  
  
const add  = (a, b) =>{*

*return a+b;*

*};*

*Const substract  = (a, b) =>{*

*return a-b;*

*};*

*// module.exports = {*

*// add:add,*

*// substract: substract*

*// };*

*module.exports = {*

*add,*

*substract*

*}*

*index.js*

*const math = require("./math");*

*console.log(math.add(2, 3));*

*console.log(math.substract(2,3));*

*4. Pattern*

*Destructuring  
After requiring the module we destructure the function from the object,  
  
math.js*

*const add  = (a, b) =>{*

*return a+b;*

*};*

*const substract  = (a, b) =>{*

*return a-b;*

*};*

*// module.exports = {*

*//     add : add,*

*//     substract : substract*

*// }*

*//if the key , value are same*

*module.exports ={*

*add,*

*substract,*

*};*

*index.js*

*const math = require("./math");*

*const {add, substract } = math;*

*console.log(add(2, 3));*

*console.log(substract(2,3));*

*5. Pattern 4*

*math.js*

*module.exports.add  = (a, b) =>{*

*return a+b;*

*};*

*module.exports.substract  = (a, b) =>{*

*return a-b;*

*};*

*Index.js*

*const math = require("./math");*

*const {add, substract } = math;*

*console.log(add(2, 3));*

*console.log(substract(2,3));*

*6. pattern 5*

*Main.js*

*exports.add  = (a, b) =>{*

*return a+b;*

*};*

*exports.substract  = (a, b) =>{*

*return a-b;*

*};*

*index.js*

*const math = require("./math");*

*const {add, substract } = math;*

*console.log(add(2, 3));*

*console.log(substract(2,3));*

[*https://www.geeksforgeeks.org/import-and-export-in-node-js/*](https://www.geeksforgeeks.org/import-and-export-in-node-js/)

*GITHUB*

[*https://github.com/anil-sidhu/node-js/tree/fundamentals*](https://github.com/anil-sidhu/node-js/tree/fundamentals) *app.js*

*module.exports={*

*x:10,*

*y:20,*

*xyz:function()*

*{*

*return 10+20;*

*}*

*};*

*index.js*

*const app=require('./app')*

*let arr=[4,1,6,3,4,89,0];*

*console.log(app.xyz())*

*const result =arr.filter((item)=>{*

*return item>=4*

*});*

*console.warn(result);*

*Node JS in Hindi #6 Core Modules  
  
github*

[*https://github.com/anil-sidhu/node-js/tree/core-modules*](https://github.com/anil-sidhu/node-js/tree/core-modules) *module - by default feature har ek programming language ke hote hai*

*two types of module*

*Core Module 1.) Global 2.) Non Global*

*External module*

*Core Modules : har ek programming language ke undar ek basic feature phle se hi hote hai jaise ..this is called core module…data base se connect krna, files vagear create krne ke liye, code ko process krne ke liye, api call krne ke liye or.. core module.. kyuki ye basic se hi hamare programming language me bane hote hai.*

*Index.js - core module - console.log(“hello world”);*

*File system, buffer, http, api calls, database connections, these are core module.*

*Global, Non Global Module*

*Global aise module hote hai jinko import krne ki jarurat nhi…*

*Global and non Global*

*Glaobal Module*

*Global aise module jinhe import krne ki jarurat nhi hoti.*

*console.log(“hello world”); //Global module*

*Non Global Module*

*jo module hume import krne pdte hai non global module.*

*Index.js*

*const fs = require("fs");*

*fs.writeFileSync("hello.txt", "hello world");*

*//write this code to create files.*

*Output :*

*Hamari file create ho jayegi : hello.txt*

*jo module hume import krne hote hai : non global module hote hai*

*jo import nhi krne padte wo hamare : Global module hote hai : console.log*

*To check directories*

*console.log('dir name', \_\_dirname);*

*node .\index.js*

*C:\Users\HP\Desktop\node-tut*

*To check filename*

*const fs = require(‘fs’);*

*console.log("filename", \_\_filename);*

*C:\Users\HP\Desktop\node-tut\index.js*

*or*

*const fs = require(‘fs’).writeFileSync;*

*fs(“abc.txt”, abc);*

*Node JS in Hindi #7 Create Server*

Github  
[*https://github.com/anil-sidhu/node-js/blob/basic-server-with-api/index.js*](https://github.com/anil-sidhu/node-js/blob/basic-server-with-api/index.js)[*GitHub - anil-sidhu/node-js at basic-server-with-api*](https://github.com/anil-sidhu/node-js/tree/basic-server-with-api)   
We have two ways to create Server using 1. Node js 2. Express js

*const http = require('http');*

*http.createServer((req,resp) =>*

*{*

*resp.write("<h1>Code Step by step</h1>");*

*resp.end();*

*}).listen(4500);*

*Node JS in Hindi #8 All About Package.json file*

*What is package file*

*Install external package*

*Package.json : ye hamre project ki detail rkhti hai : coding se related : jaise hamare projects ka version kya hai: uska name kya hai: uski git repository kya hai : uske undar koun koun si command use ho rhi hai : uske undar koun koun se package dal rhe hai : jab ek project bnate hai to usme bahut sare projects bna rkhe hai : validation ke liye hmne dal liya : kuch console logger wagera dal liya data base connection ke liye : mango db :*

*mysql : external package wo sare ke sare kaise manage hote hai : wo hote hai : package.json me manage hote hai*

*Kaise project build bnati hai*

*Kaise project run krni hai*

*Command run krni hai*

*Ye sare package.json me hoti hai*

*Package.json hamare projects ki details rakhti hai*

*coding se related details rkhti hai*

*Index.js*

*console.log(“package.json”);*

*npm init : enter : package.json file ban jayegi :*

*package.lock.json : package ki detail rkhera*

*node js single threaded hai ya multi threaded*

*node js single threaded hai : ek time me ek hi command run krti hai*

*multi threaded me ke sath : 4,5 command ko ek sath run kr skti hai*

*Node JS in Hindi #9 A challenge and 2 important interview question*[*GitHub - anil-sidhu/node-js at challange-1*](https://github.com/anil-sidhu/node-js/tree/challange-1) *node module galti se delete ho gya : npm install*

*Node JS in Hindi # 10 All About Nodemon package  
  
nodemon time saving module*

*es package ka use krke bahut sara time save kr skte hai : sare project me time save krne ke liye use kiya jata hai   
  
installation nodemon : nodemon documentation : cmd : npm I nodemon -g*

*to run : folder directories: : nodemon .index.js  
  
ab code me kuch bhi change krenge to apne app run hoga : time saving*

*interview question : node js : async language : node js single threaded*

*sync : lets see 3 4 syntax run kr di : to wo sync me chlenge : 1st, 2nd, 3rd, 4th*

*async : if second syntax is taking time for fun : will execute next one : then 2nd one will execute wait nhi kr skti kisi bhi script ka*

*Node JS in Hindi # 11 Simple and basic API*[*GitHub - anil-sidhu/node-js at basic-api*](https://github.com/anil-sidhu/node-js/tree/basic-api) *make a server*

*create header and api body*

*create an api with static data*

*put data in api file*

*data.js  
const data=[*

*{name:'anil',mail:'anil@test.com'},*

*{name:'sam',mail:'peter@test.com'},*

*{name:'peter',mail:'peter@test.com'}*

*]*

*module.exports=data;*

*index.js*

*const http= require('http');*

*const data = require('./data');*

*http.createServer((req,resp)=>{*

*resp.writeHead(500,{'Content-Type':'application\json'});*

*resp.write(JSON.stringify(data));*

*resp.end();*

*}).listen(5000);*

*Node JS in Hindi # 12 Getting input from command line*

[*GitHub - anil-sidhu/node-js at cmd-input*](https://github.com/anil-sidhu/node-js/tree/cmd-input) *set input from command line*

*create file with input*

*delete the file with input*

*index.js*

*console.log(“code step by step”);*

*run - node index.js*

*code step by step*

*ab maine node js ke sath me kuch extra parameter pass kr diya*

*node index.js 100 anil*

*ye jo input hai usko node js ke undar use kr paye*

*abhi kuch nhi hoga normal output hai whi chlegi 100 anil jo hai - esi ko hum get krenge*

*index.js*

*console.log(process);*

*process ek object hota hai jiske undar almost sab kuch hota hai node js ka : run krne me kitna sari chige print ho jayega.*

*console.log(process.argv)3*

*ye yha pe 2 properties de di*

*two input*

*first node js ki hamari file save hai wo*

*'C:\\Program Files\\nodejs\\node.exe',*

*second - jaha se hum esko run kr rhe hai wo*

*'C:\\Users\\kamlesh\\Desktop\\4. node js\\3. node js practice\\index'*

*Terminal - node index.js hello*

*ab 3 input aa gya*

*'C:\\Program Files\\nodejs\\node.exe',*

*'C:\\Users\\kamlesh\\Desktop\\4. node js\\3. node js practice\\index'*

*Hello*

*Terminal - node index.js hello hi*

*ab 4 input aa gya*

*'C:\\Program Files\\nodejs\\node.exe',*

*'C:\\Users\\kamlesh\\Desktop\\4. node js\\3. node js practice\\index'*

*Hello*

*Hi*

*Ab ek ek krke kaise get kr skte hai*

*Console.log(process.argv[2]);*

*Output*

*Hello*

*Console.log(process.argv[3]);*

*Hi*

*first-properties, second properties, hello*

*Console.log(argument.argv[2]) - as an array*

*Scenario - add/remove - name – text*

*index.js*

*to create file - we need file system module- jo ki node js ke undar inbuild hota hai*

*const fs = require(‘fs’);*

*const input = process.argv;*

*fs.writeFileSync(input[2], input[3]);*

*1st : input[2] : file ka name*

*1st : input[3] : data jo kuch hum enter krna chahte hai*

*Terminal : node index.js apple.txt ‘this is a fruits’*

*Output : jaise hi enter kiya apple.txt file create ho gyi*

*and text : this is a fruit*

*const fs = require(‘fs’);*

*const input = process.argv;*

*fs.writeFileSync(input[2], input[3]);*

*scenorio : ab esko hum jahte hai : add bhi kr pau or remove bhi kr pau*

*const fs = require(‘fs’);*

*const input = process.argv;*

*fs.writeFileSync(input[2], input[3]);*

*If(input[2] = = ‘add’){*

*fs.writeFileSync(input[3], input[4])*

*} else if (input[2] == ‘remove’) {*

*fs.unlinkSync( input[3])*

*}*

*else{*

*console.log(“invalid output”);*

*}*

*run : node index.js add orange.txt “this is color and fruit”*

*orange.txt - name se file ban gyi - or text - this is color and fruit*

*lets see ab file ko remove krna chahte*

*hai*

*run : node index.js remove orange.txt - hamara orange.txt file remove ho jayega*

*Node JS in Hindi # 12 Getting input from command line*

*Node JS in Hindi # 13 Display file list from folder*[*GitHub - anil-sidhu/node-js at file-list*](https://github.com/anil-sidhu/node-js/tree/file-list)

*Node JS in Hindi # 14 Asynchronous Programming Language  
  
crud with file system*

*make file*

*read file*

*update file*

*rename file*

*delete file*

[*GitHub - anil-sidhu/node-js at curd*](https://github.com/anil-sidhu/node-js/tree/curd)

*Node JS in Hindi # 15 Asynchronous Programming Language*

[*GitHub - anil-sidhu/node-js at async*](https://github.com/anil-sidhu/node-js/tree/async)

*Node JS in Hindi # 16 Handle Asynchronous Data*[*GitHub - anil-sidhu/node-js at handle-async*](https://github.com/anil-sidhu/node-js/tree/handle-async) *Node JS in Hindi # 17 How Node js Works*[*GitHub - anil-sidhu/node-js at node-js-work-process*](https://github.com/anil-sidhu/node-js/tree/node-js-work-process) *Node JS in Hindi # 18 Express js Introduction*[*GitHub - anil-sidhu/node-js at express*](https://github.com/anil-sidhu/node-js/tree/express)

*Node JS in Hindi # 19 Routing Params - Request and Response*[*GitHub - anil-sidhu/node-js at express*](https://github.com/anil-sidhu/node-js/tree/express)

*Node JS in Hindi # 20 Render HTML and JSON*[*anil-sidhu/node-js at html-json-render (github.com)*](https://github.com/anil-sidhu/node-js/tree/html-json-render)

*Node JS in Hindi # 21 Make HTML page*

[*GitHub - anil-sidhu/node-js at make-html-page*](https://github.com/anil-sidhu/node-js/tree/make-html-page)

*Node JS in Hindi # 22 Remove extension from URL and 404 Page*

[*anil-sidhu/node-js at 404-page (github.com)*](https://github.com/anil-sidhu/node-js/tree/404-page)

*Node JS in Hindi # 23 EJS Template Engine*[*node-js/index.js at ejs · anil-sidhu/node-js (github.com)*](https://github.com/anil-sidhu/node-js/blob/ejs/index.js#L3)

*EJS (Embedded JavaScript) is a popular templating engine for Node.js that allows you to generate HTML markup with plain JavaScript. It is particularly useful for creating dynamic web pages, as it enables you to embed JavaScript logic directly within your HTML.  
  
EJS is a simple templating language that is used to generate HTML markup with plain JavaScript. It also helps to embed JavaScript into HTML pages.  
connect mongodb with node js*[*https://www.w3schools.com/nodejs/nodejs\_mongodb.asp*](https://www.w3schools.com/nodejs/nodejs_mongodb.asp) *error :  
How to Fix Mongoose Server Selection Error connect ECONNREFUSED*[*https://youtu.be/ANeOfRDPOfw*](https://youtu.be/ANeOfRDPOfw) *const url= 'mongodb://127.0.0.1:27017';  
  
npm install mongodb*

*var mongo = require('mongodb');*

SATARTING THE DOCUMENTATION  
 *Github – code step by step – go through every github branch*[*Branches · anil-sidhu/node-js (github.com)*](https://github.com/anil-sidhu/node-js/branches/stale?page=1)

*response in express Js  
resp.send();  
resp.sendFile();*

*resp.render();  
  
app.set(‘view engine’, ‘ejs’)  
  
Middleware*

*important topics  
Crud with Mongoose*[*anil-sidhu/node-js at curd-with-mongoose (github.com)*](https://github.com/anil-sidhu/node-js/tree/curd-with-mongoose)

*index.js  
  
connect with db using mongoose  
const mongoose = require(‘mongoose’);  
mongoose.connect(“mongodb urls”);  
  
//creating schema  
const productSchema = new mongoose.shema({  
name : String,*

*price : String,  
brand : String,   
category : String   
});  
  
//Operation – CRUD  
save data in database,   
update data,   
delete data,   
find data  
  
//save data in database   
using asynchronous function   
  
const saveinDB = async () => {  
const Product = mongoose.model(‘products’, productSchems);  
const data = new Product({  
name: "max 100",*

*price: 200,  
brand: 'maxx',*

*category: 'Mobile'  
});  
const result = await data.save();  
}   
  
//update db  
const updatedb = async () => {*

*const Product = mongoose.model(“products”, productSchema);  
const data = await Product.updateOne(  
{ name: "max 6" },*

*{*

*$set: { price: 550,name:'max pro 6' }*

*}  
)*

*console.log(data);  
}  
  
//delete data   
const deletedb = async () => {*

*const Product = mongoose.model(‘products’, productSchema);  
const data = await Product.deleteOne(({name:'max 100'}));  
  
console.log(data);*

*}*

*//finddb  
const findDB = async () => {*

*const Product = mongoose.model(‘products’, productSchema);  
const data = await Product.find({name:'max pro 611'});  
console.log(data);*

*}*[*node-js/index.js at insert-api-mongose · anil-sidhu/node-js (github.com)*](https://github.com/anil-sidhu/node-js/blob/insert-api-mongose/index.js) *important*[*node-js/index.js at insert-api-mongose · anil-sidhu/node-js (github.com)*](https://github.com/anil-sidhu/node-js/blob/insert-api-mongose/index.js)

*req.body -   
req.query  
req.params  
The****req.body****property contains key-value pairs of data submitted in the request body. By default, it is undefined and is populated when you use a middleware called body-parsing such as express.urlencoded() or express.json().   
  
Param – parameter****Express: req.params, req.query and req.body***

[*https://dev.to/gathoni/express-req-params-req-query-and-req-body-4lpc*](https://dev.to/gathoni/express-req-params-req-query-and-req-body-4lpc)

*Express js Highlight   
app.get – send, sendFile, render(ejs);  
app.use – middleware function , route,   
app.use(express.static)- to read static file  
app.use(express.json()); - body parsing to populate date in req.body , // For parsing application/json  
req.body  
  
Mongo DB – mongoose – highlight  
cerate DB - new Product . we need to provide object here in a key , valus pair means as an object  
updateOne – which one object we are going to update pluuus on that selected object what are the things we are going to update  
deleteOne – give the object name we are going to delete.  
findDb – find – inside find give object in a key, value pair*

***Code step by step   
25, 26  
25, 26 video is very important   
esme route middleware ke bare me btaya gya hai***

//route, middleware - next (), application middleware, route level middleware

1. route example

const express = require("express");

const app = express();

app.get('/', (req, resp) => {

  resp.send("welcome to home page");

});

app.get("/about", (req,resp) => {

resp.send("welcome to home page");

});

2. next(); middleware

const express = require('express');

const app = express();

const reqFilter = (req, resp, next) => {

  if(!req.query.age){

    resp.send("please enter your age");

  }

else if (req.query.age<18){

resp.send("you are under aged");

}

else{

  next();

}

};

app.use(reqFilter);

app.get("/",(req, resp) => {

resp.send("welcome to home page");

});

app.get("/about", (req, resp) => {

resp.send("welcome to home page");

});

output :

ye sabhi route pe apply honge home and about pe bhi or agar koi or bhi route ho usme bhi, we can add other route as well , it will apply on every route

3. another way to write , this middleware will apply on speacific route

const express = require();

const app = express();

const reqFilter = (req, resp, next) => {

if (!req.query.age){

  resp.send("please enter your age");

}

else if(req.query.age<18){

resp.send("you are under aged");

}

else{

  next();

}

};

app.use(reqFilter);

//middleware - applying middleware in a home page only - this is called application level middleware

app.get("/", reqFilter, (req, resp) => {

resp.send("welcome to home page");

});

app.get("/", (req, resp) => {

  resp.send("welcome to home page");

  });

  //middleware - applying middleware in a home page only - this is called application level middleware

  es application level middleware ko 1 route, group of

  4. route level middleware

  const express = require('express');

  const app = express();

  const reqFilter = (req, resp, next) => {

    if (!req.query.age){

      resp.send("please enter your age");

    }

    else if(req.query.age<18){

    resp.send("you are under aged");

    }

    else{

      next();

    }

    }

const route = express.Router();

route.use(reqFilter);

//route level - middleware - here applying only on home pages

route.get("/", (req, resp) => {

resp.send("welcome to home page");

  });

  app.get("/about", (req, resp) => {

    resp.send("welcome to about page");

      });

app.use('/', route);

node js middleware  
<https://youtu.be/6Spfry-_fAY?list=PL8p2I9GklV456iofeMKReMTvWLr7Ki9At>

<https://youtu.be/2e2E5cvLw0g?list=PL8p2I9GklV456iofeMKReMTvWLr7Ki9At>

In Node.js, middleware refers to a function or set of functions that are executed in a specific order during the processing of an HTTP request. Middleware functions have access to the request and response objects, and they can perform various tasks such as modifying the request or response, executing additional code, or passing control to the next middleware function in the stack.

Middleware functions in Node.js can be used for a variety of purposes, such as handling authentication, logging, error handling, request parsing, and more. They provide a way to modularize and separate concerns in your application.

Here's an example of how middleware can be used in a Node.js application:

const express = require('express');

const app = express();

// Middleware function

const logger = (req, res, next) => {

  console.log(`[${new Date().toISOString()}] ${req.method} ${req.url}`);

  next(); // Call the next middleware function

};

// Registering the middleware

app.use(logger);

// Route handler

app.get('/', (req, res) => {

  res.send('Hello, World!');

});

// Start the server

app.listen(3000, () => {

  console.log('Server is listening on port 3000');

});

In the above example, we're using the Express framework, which is a popular framework for building Node.js applications. The **logger** function is a middleware function that logs the timestamp, HTTP method, and URL of each incoming request. We register the middleware using **app.use(logger)**, which means that it will be executed for every incoming request before reaching the route handler.

Middleware functions can also be used selectively for specific routes by specifying the route as an argument to **app.use** or **app.METHOD** (e.g., **app.get('/users', logger, getUsers)**). Additionally, multiple middleware functions can be chained together by passing them as arguments to **app.use** or **app.METHOD**.

This is just a basic example, and middleware can be customized and expanded to suit the needs of your application. Middleware plays a crucial role in Node.js applications by providing a flexible way to handle common tasks in a modular and reusable manner.  
  
*@@@@@@@@@@@@@@@@@@@@@@@@*[*Sheryians Coding School*](https://www.youtube.com/@sheryians)

***Express.js - Learn What Matters: Mastering the Framework | Backend (Node JS) Series***

[***https://www.youtube.com/watch?v=jhihI4kBAN8&list=PLbtI3\_MArDOk7J-8hR6CeB5U6bvgRKNNr&index=3***](https://www.youtube.com/watch?v=jhihI4kBAN8&list=PLbtI3_MArDOk7J-8hR6CeB5U6bvgRKNNr&index=3)

[***https://youtu.be/pKJ4GGyDgJo?si=KaOgAXSSE0bwvtuL***](https://youtu.be/pKJ4GGyDgJo?si=KaOgAXSSE0bwvtuL)

[***https://youtu.be/T55Kb8rrH1g?si=lZjEEXDA9IHa5SDh***](https://youtu.be/T55Kb8rrH1g?si=lZjEEXDA9IHa5SDh)

***1.*** [***https://www.youtube.com/watch?v=jhihI4kBAN8&list=PLbtI3\_MArDOk7J-8hR6CeB5U6bvgRKNNr&index=3***](https://www.youtube.com/watch?v=jhihI4kBAN8&list=PLbtI3_MArDOk7J-8hR6CeB5U6bvgRKNNr&index=3) ***install node js   
import export in node js  
install any package using npm  
  
  
2.*** [***https://youtu.be/pKJ4GGyDgJo?si=KaOgAXSSE0bwvtuL***](https://youtu.be/pKJ4GGyDgJo?si=KaOgAXSSE0bwvtuL)

***Node js , express js   
routing  
middleware  
req, resp  
route , parameter  
template  
static files   
http method – get and post  
error handling  
  
Routing -   
routing means urls ,   
localhost:5000/about***npm I express  
[***express - npm (npmjs.com)***](https://www.npmjs.com/package/express)