**Node JS**

*//script.js*

const a=20;

const b=30;

setTimeout(() => {

console.log(a+b);

},3000 );

*// node script.js*

Allows JS to run on machines rather than browser

Install node js LTS version.

In cmd ,

Node - v , to check the version of node / whether it is installed

Node (type node and enter) , node cmd prompt opens

global // it lists all the keywords available to use with node (does not include browser properties and methods)

Like Window obj in browser, node has global

**Gitbash /cmd**

* **echo >** *script.js* - to create new file
* **out-File** script.js **-Encoding utf8**
* **node** script.js - to execute file using node

**node**

* **console.clear() –** to clear the console
* **process -** details of current process
* **process.exit() –** to exit the current process
* **\_ \_ dirname –** current directory

Node does not support import statements [ES6] . Hence use the older method modules

Kinds of Modules:

1. User Moduels

Script1.js

const largeNumber = 356;

**module.exports = {**

*largeNumber:largeNumber,*

**}**

// export default largeNumber

Script2.js

**const obj=require('./script1.js');**

*//import largeNumber from ‘script.js’*

const a **= obj.largeNumber**;

const b = 10;

console.log(a+b);

*// node script2.js ( to execute)*

1. Node - Built in Modules

fs : **const obj=require(‘fs’);** //file system

http : **const obj=require(‘http’);** //to build server

1. Modules / packages from npm

**npm init –y** // to create package .json

**npm install --save-dev nodemon**

***//*** *in json script type – “start”:”nodemon”*

*// to execute another file – “start” :”nodemon script2.js”*

**Building a server**

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

const server = **http.createServer**((request,response)=>{

console.log(**request.headers**);

console.log(**request.method**);

console.log(**request.url**);

**response.setHeader('content-type','text/html');**

**response.end('<div> This is a server response ;-) </div>');**

});

server.listen(3000);

const http = require ('http');

const server = http.createServer((request,response)=>{

console.log("Movie - This week ");

**const user = {**

**name:'NGK',**

**date:'31-05-2019',**

**actor:'Surya',**

**AGE:38,**

**}**

const mydata = request.user-agent;

response.setHeader(**'content-type','application/json'**);

**response.end(JSON.stringify(user));**

});

server.listen(3000);

**Introduction to express.js**

**npm install express** // in working directory

It is a library for the backend.

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

**const app = express();**

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

**res.send("helooo");**

});

const user = {

name:'NGK',

date:31,

actor:'Surya',

}

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

res.send(user);

});

app.listen(3000);

**Express Middleware**

Middleware receives the request first (it may be used to modify) and passes through express codes when **next()** is called

**app.use((req,res,next)=>{**

console.log("Hello");

**next();**

});

**Postman**

It’s a tool that allows us to make these get and post requests to server and see what we get back

//to test more than the get request and to make post request to work

[www.getpostman.com](http://www.getpostman.com)

Download, install and signup

After executing npm start (starting server) -> in post man enter the url => localhost:3000 and press send .

To access the information in the body 🡪 add ‘console.log(req.body)’ inside post function and click send from postman

You get -> undefined . // express can give only few things. To access this you need to use **middleware** called -> **body-parser**.

*npm install* ***body-parser***

**const bodyParser = require('body-parser');**

**app.use(bodyParser.urlencoded({extended:false}));**

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

**console.log(req.body);**

res.send(user);

});

**To send Key value with form data**

**// if you build a server for the form that gets submitted we use x-www-form-urlencoded**

To send json

In postman click on ‘raw’ and in ‘text’ select ‘JSON’ and then type

{

"user":"Surya",

"profession":"Actor in movies"

}

Click send // you will get empty object because parser was earlier asked to parse urlencoded . so add

*app.use(bodyParser.json());*

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

console.log(req.body);

**res.send(‘success’)// res.send(req.body);**

});

RESTful APIs

Rules that allows browser and server to communicate through functions like get,put,post,delete

They are called **stateless: independent calls and it has all the data to complete itself.**

app.get('/:id',(req,res)=>{ // id is used to get parameter value like locahost:3000/*123*

**console.log(req.query); //** gets the query strings

**console.log(req.body); //**gets the value passed through body

**console.log(req.headers); //** gets the header values

**console.log(req.params); //** gets the parameters

res.send("hi server has started ");

})

res.status(404).send('not found'); // response with status

To send static files like html , css ,js

Create a html file and store it in public folder. then

app.use(express.static(\_\_dirname +'/public'));

**Node File System Module**

**readFile :** Asynchronous method . Returns the control while reading

**readFileSync** : Synchronous method. Control has to wait till it finishes reading

const fs = require('fs'); // fs – file system

fs.readFile('./hello.txt',(err,data)=>{

if(err){

console.log('errrorrr');

}

else{

console.log('Async :' ,data.toString('utf8'));//utf8 – encoding format advanced that ascii

}

})

const file =fs.readFileSync('./hello.txt');

console.log('Sync :' ,file.toString()); // execute using - node filename.js

**appendFile :** adds the content to the existing file or creates a new file and adds the content

fs.appendFile('./hello.txt',' This is actually cool!!!',err=>{

if(err){

console.log(err);

}

})

writeFile : create a new file and write content into it

fs.writeFile('bye.txt','Sad to see me suffer with this dizzy health:-( ', err =>{

if(err){

console.log(err);

}

})

Unlink (delete file) :

fs.unlink('./bye.txt',err=>{

if(err){

console.log(err)

}

console.log("file deleted");

});

Time calculation for code execution:

fs.readFile('./hello.txt',(err,data)=>{

**console.time('fun'); //** label fun

if(err){

console.log('errrorrr');

}

else{

console.log('Async :' ,data.toString('utf8'));

}

**console.timeEnd('fun'); //** same label should be given

})

Const directionsArray = directions**.split();** // directions is a text file . when splitting it is made into array

Const answer = directionArray.***some(value) //***some looks through the array step by step and once it finds the match it stops looping the array

**Creating a server for myfacerecognition**

Creaete a new folder – smart-brainapi

date ***= new Date()***

Create package.json using***npm init***

***npm insall express***

***npm install body-parser***

***npm install --save-dev nodemon***

***npm install bcrypt // to encrypt passwords***

*sensitive data : https , post and generate hash*

**echo > server.js**  // create a server file

in package.json - > scripts -> “start”:”nodemon server.js”

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

*const bodyparser = require('body-parser');*

*const app = express();*

***app.use(bodyparser.json());***

*const database = {*

***users:[*** *{*

*userid :123,*

*username :'john',*

*email : 'john@gmail.com',*

*password : 'password',*

*entries : 0,*

*join :* ***new Date(),***

*},*

*{*

*userid :124,*

*username :'sally',*

*email : 'sally@gmail.com',*

*password : 1234,*

*entries : 0,*

*join :* ***new Date(),***

*}*

***]***

*};*

*app.get('/',(req,res)=>{ // root api GET*

***res.json(database.users);***

*})*

*// singin api POST*

*app.post(‘****/signin’****,(req,res)=>{*

***if(req.body.email === database.users[0].email && req.body.password === database.users[0].password){***

***res.json(‘signin successful’);***

*}*

*else{*

*res.****status(400)****.json(‘error in signin’);*

*}*

*})*

*//register api POST*

*app.post****(‘/register’****,(req,res)=>{*

***const {name,email,password} = req.body;***

***database.users.push(***

*{*

*id : 125,*

*name : name,*

*email : email ,*

*password : password,*

*entries : 0,*

*join : new Date(),*

*}*

*);*

***res.json(database.users[database.users.length-1]);***

*})*

***app.listen****(3000,()=>{*

*console.log(“app is running on port 3000”);*

*});*

*// profile/:userid api GET*

*app.get(‘****/profile/:id’****,(req,res)=>{*

***const {id} = req.params;***

*let found = false;*

*database.users.forEach(user=>{*

*if(user.id == id){*

*found = true;*

*return res.json(user);*

*}*

*})*

*if(!found){*

*res.status(404).json(‘user not found’);*

*}*

*})*

*// image api PUT*

***app.put('/image'****,(req,res)=>{*

*const {id} = req.body;*

*let found = false;*

*database.users.forEach(user => {*

*if(user.id == id){*

*found = true;*

*user.entries++;*

*return res.json(user.entries);*

*}*

*})*

*if(!found){*

*return res.json('user not found');*

*}*

*})*