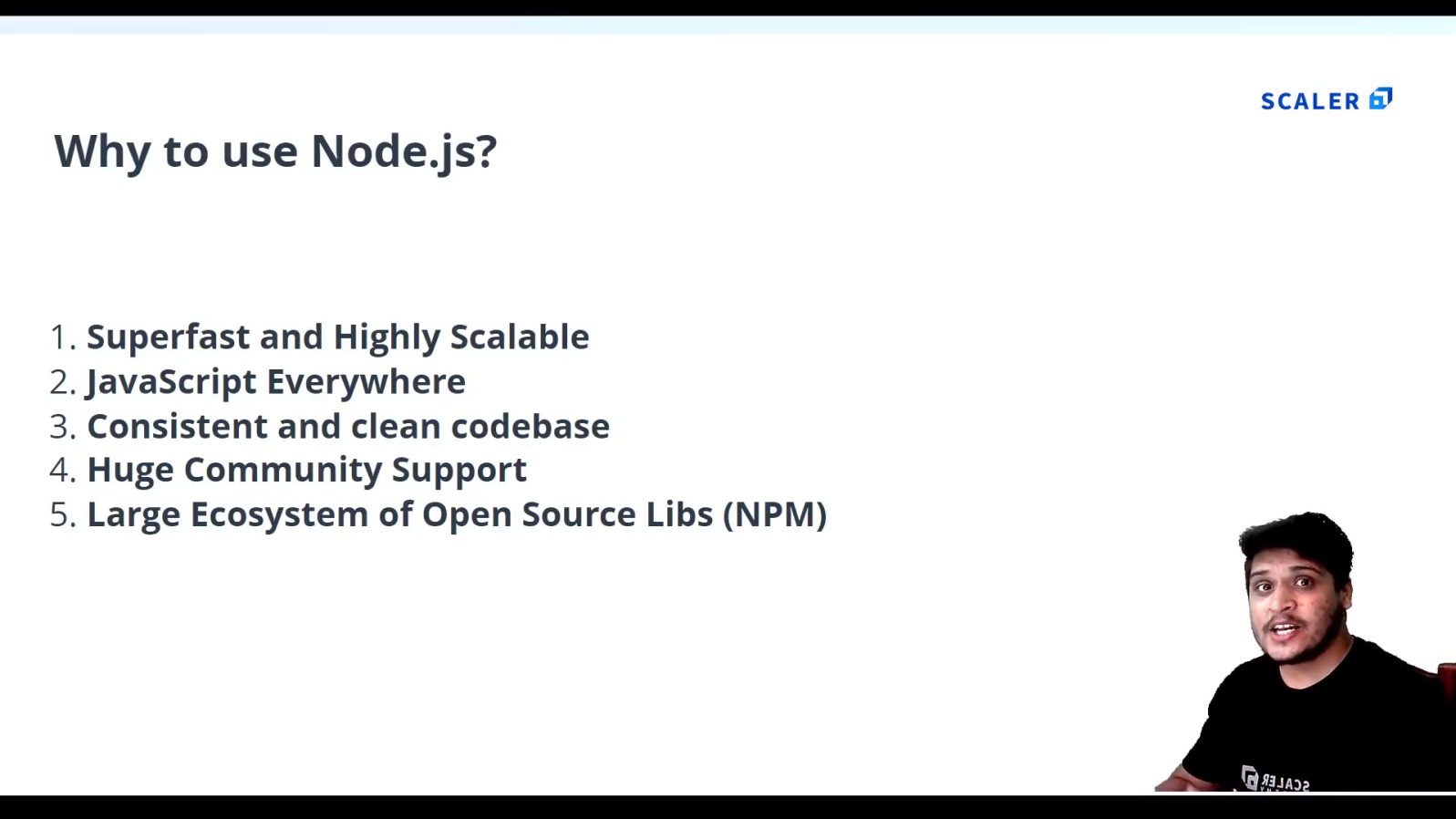
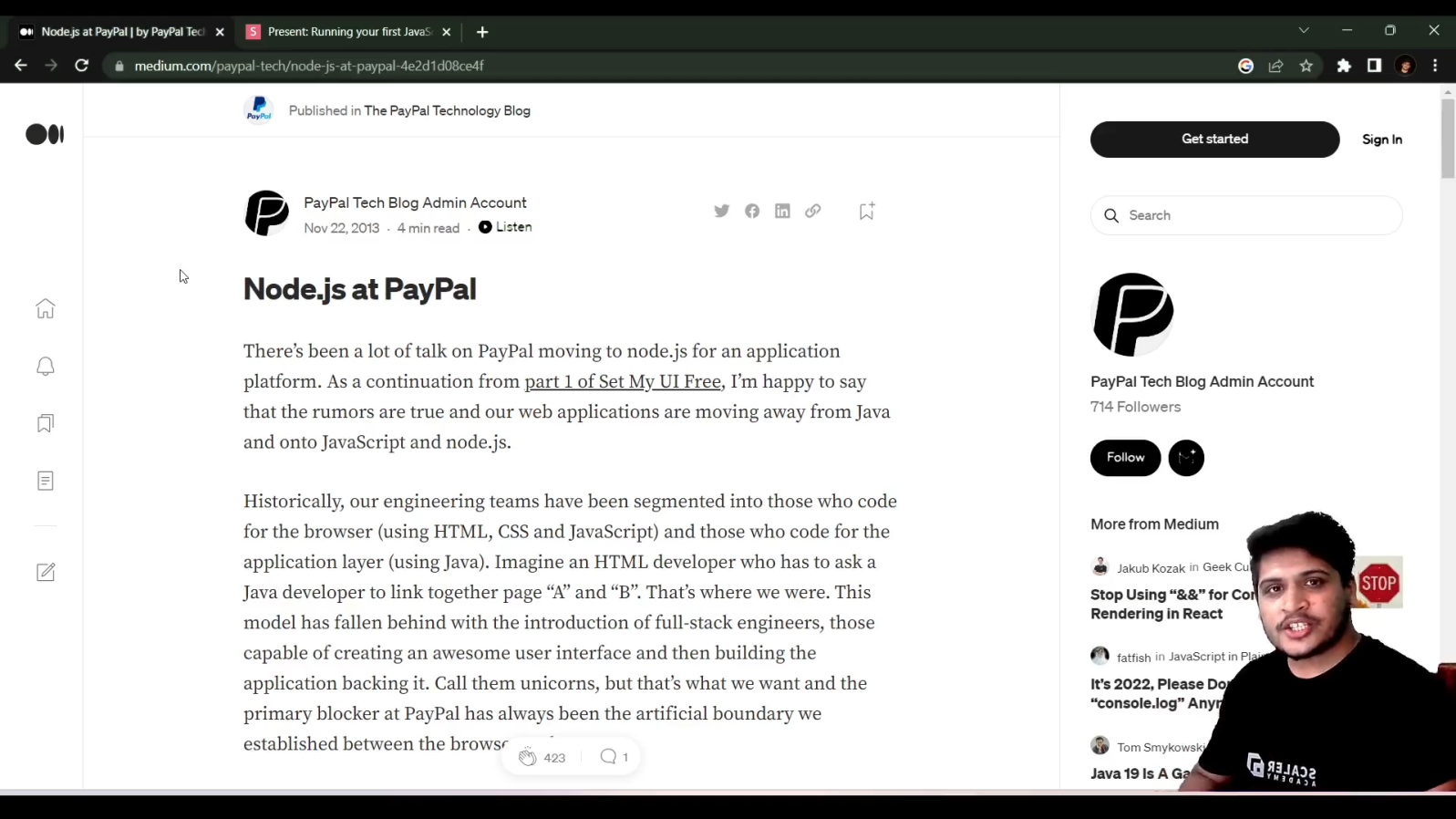
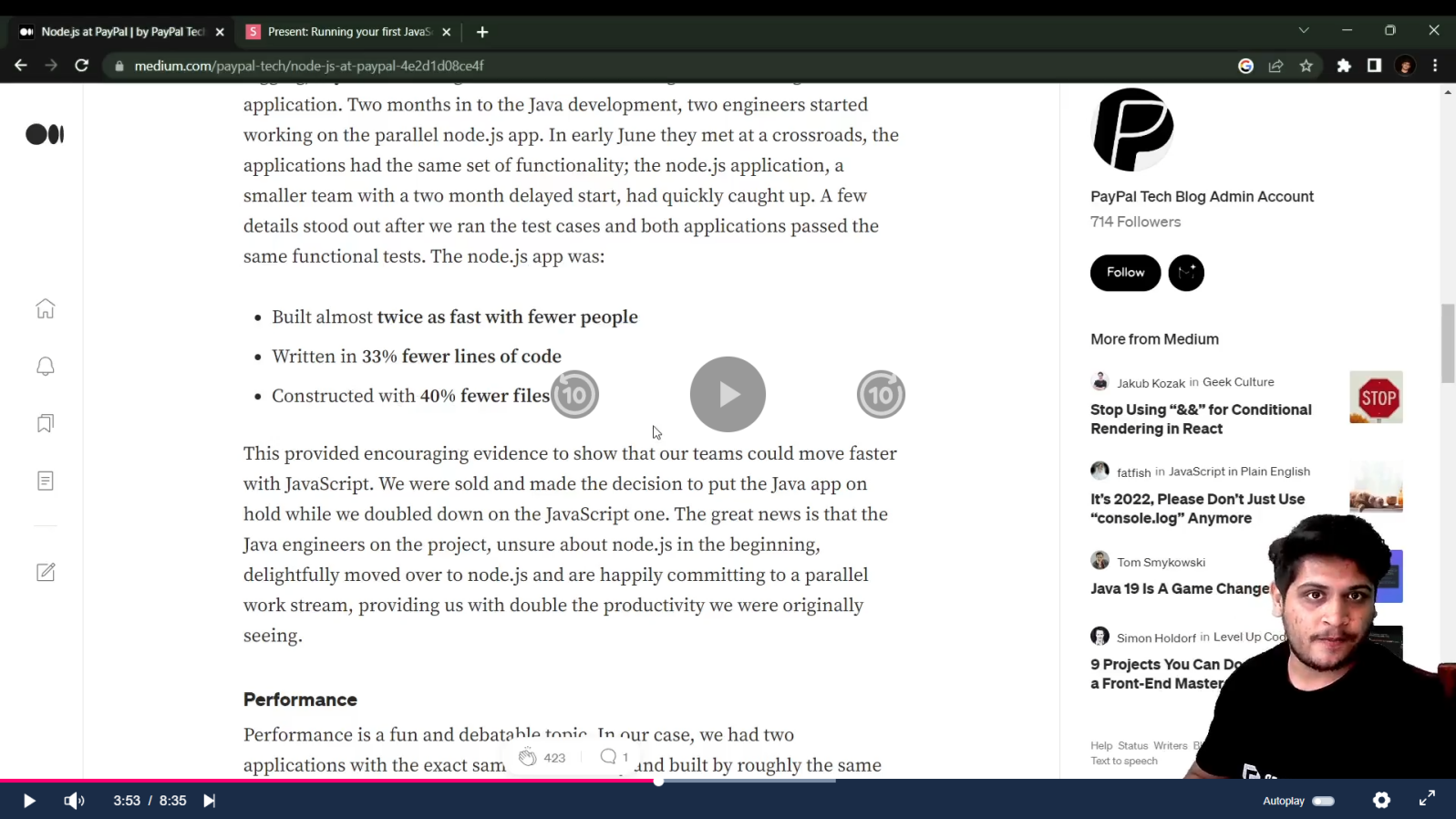
**NODE JS - SCALAR**

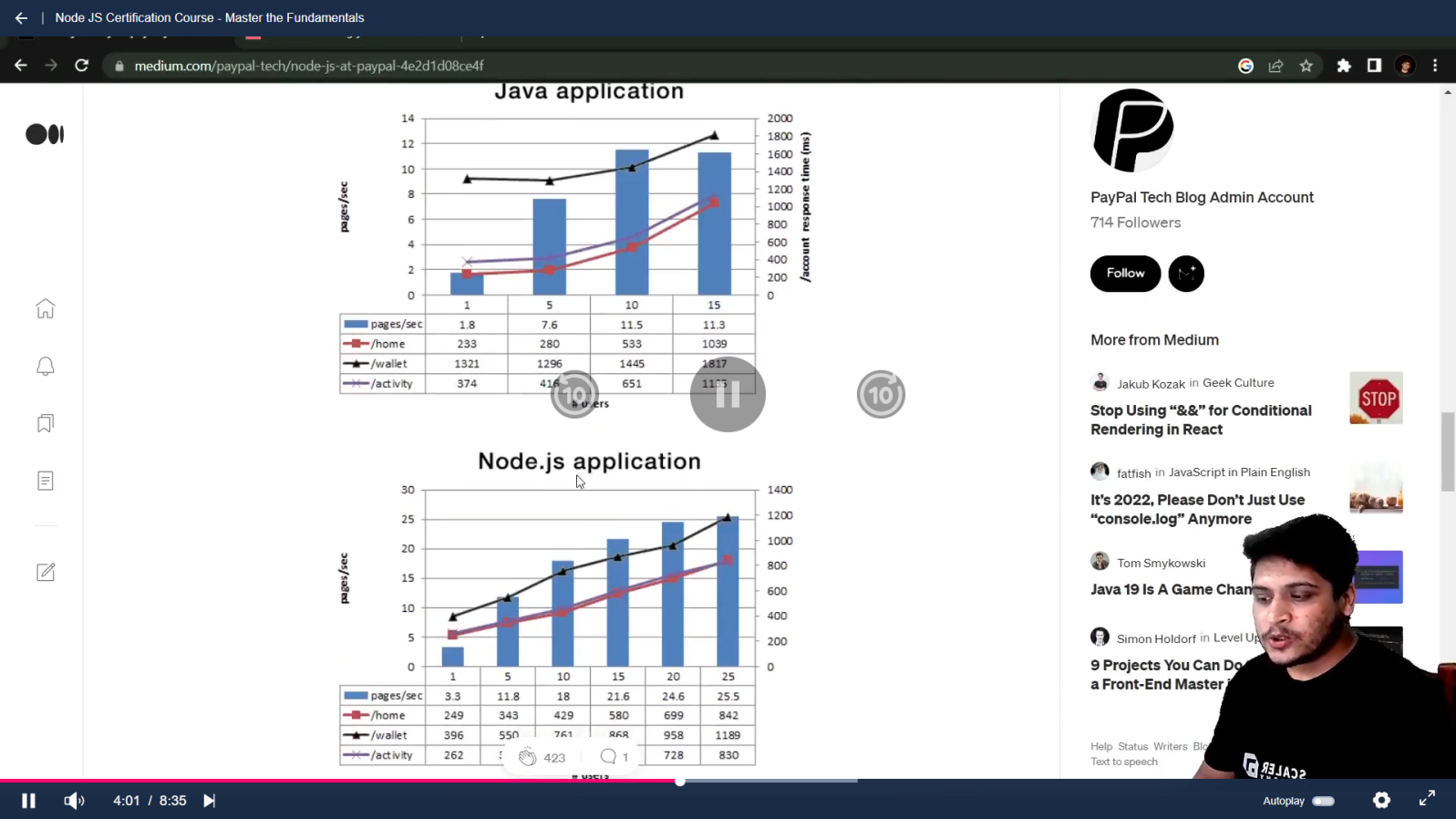
1. **Introduction :**

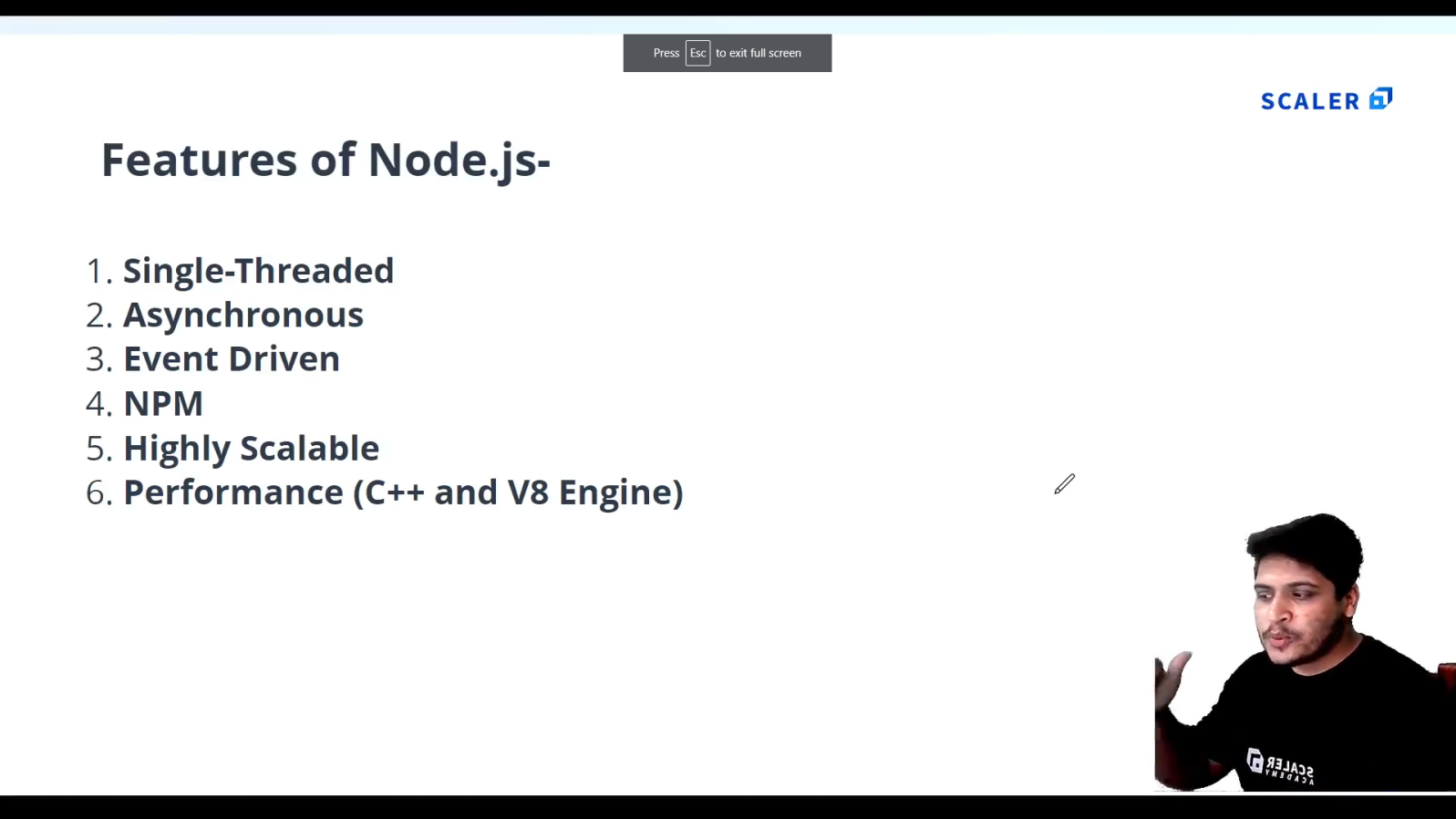
****

****

****

****

****

****

1. **Install Node.js and VScode.**
2. **First Node program :**
   1. **Create a js file with a function:p1\_test.js**

console.log('Hello World');

function sayHello(){

    console.log('Hello from anshad');

}

sayHello();

* 1. **To Execute type “node name\_of\_program.js” : node p1\_test.js**

1. **There is no “window” object in Node js instead we have “global” object.**

console.log(global);

PS E:\MCA\COURSES\NODE JS\nodejs\_certification@scalar> node p1\_test.js

<ref \*1> Object [global] {

  global: [Circular \*1],

  clearImmediate: [Function: clearImmediate],

  setImmediate: [Function: setImmediate] {

    [Symbol(nodejs.util.promisify.custom)]: [Getter]

  },

  clearInterval: [Function: clearInterval],

  clearTimeout: [Function: clearTimeout],

  setInterval: [Function: setInterval],

  setTimeout: [Function: setTimeout] {

    [Symbol(nodejs.util.promisify.custom)]: [Getter]

  },

  queueMicrotask: [Function: queueMicrotask],

  structuredClone: [Function: structuredClone],

  atob: [Getter/Setter],

  btoa: [Getter/Setter],

  performance: [Getter/Setter],

  fetch: [Function: value],

  crypto: [Getter]

}

1. **Node Module system:**
   1. **The “global” Object :**

//GLOBAL OBJECT

//are built-in objects that are part of the JavaScript and can be used directly in the application without importing any particular module.

let name = 'Anshad';

console.log(global.name);//undefined

* 1. **Modules and Modularity :**

1. **Create ‘calculator.js’ file :**

//CALCULATOR :

function add(a , b ){

    console.log( a + b );

}

function sub(a , b ){

    console.log( a - b );

}

function mul(a , b ){

    console.log( a \* b );

}

function div(a , b ){

    console.log( a / b );

}

//Exporting functions:

module.exports = {

    addition : add ,

    subtraction : sub,

    multiplication : mul,

    division : div

}

1. **Create main file ‘modularity.js’ file :**

//1.create a seperate file 'calculator.js'.

//Modularity lets us use those contents in calculator.js in this file.

const calculator = require('./calculator');//import calculator.js

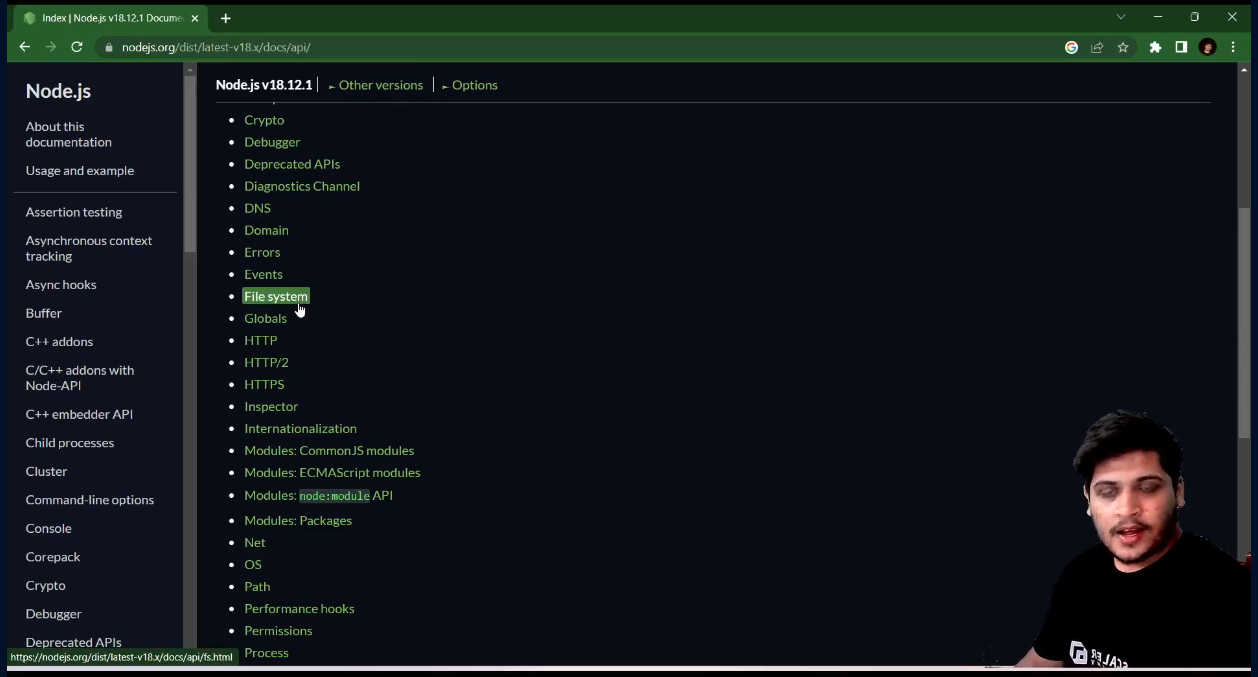
calculator.addition(3 , 4);//Calls the function in calculator.js by passing values to that.

calculator.subtraction( 5 ,2);

calculator.multiplication( 3 , 4);

calculator.division(10 , 2);

* 1. **Introduction to Node Modules :**



* 1. **Child Process Module :**

//Child process is a node module used to create sub process within a script .

//You can perform different tasks with your script by just using some methods.

const cp = require('child\_process');//Importing child-process module.

cp.execSync('calc') ; //Opens a calculator.['calc' is short code for opening calculator.]

cp.execSync('start chrome'); //opens chrome browser.

cp.execSync('start chrome https://www.google.com/');//opens google page directly

console.log('Output '+cp.execSync('node p1\_test.js'));//prints output from a specified File.

* 1. **OS Module :**

|  |  |
| --- | --- |
| **Code** | **Output** |
| //OS Module :(to get the information of your current system)  const os = require('os');//Import OS module  console.log(os.arch());//Displays Architecture of your OS (64 or 32)  console.log(os.platform());//Displays the platform of your OS  console.log(os.networkInterfaces());//Displays the network informations of your Os  console.log(os.cpus());//Displays the cpu's details (graphics,processor,configurations)  console.log('Total Memory : ',os.totalmem());//Displays the total memory .  console.log('Free Memory : ',os.freemem());//Displays the free memory. | PS E:\MCA\COURSES\NODE\_JS\nodejs\_certification@scalar\Module\_3> node .\osModule.js  x64  win32  {  Ethernet: [  {  address: '192.168.1.12',  netmask: '255.255.255.0',  family: 'IPv4',  mac: 'f8:0d:ac:29:a8:91',  internal: false,  cidr: '192.168.1.12/24'  }  ],  'Loopback Pseudo-Interface 1': [  {  address: '::1',  netmask: 'ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff',  family: 'IPv6',  mac: '00:00:00:00:00:00',  internal: true,  cidr: '::1/128',  scopeid: 0  },  {  address: '127.0.0.1',  netmask: '255.0.0.0',  family: 'IPv4',  mac: '00:00:00:00:00:00',  internal: true,  cidr: '127.0.0.1/8'  }  ]  }  [  {  model: 'AMD Ryzen 5 3550H with Radeon Vega Mobile Gfx ',  speed: 2096,  times: { user: 101765, nice: 0, sys: 115328, idle: 1025421, irq: 13640 }  },  {  model: 'AMD Ryzen 5 3550H with Radeon Vega Mobile Gfx ',  speed: 2096,  times: { user: 73468, nice: 0, sys: 51812, idle: 1117078, irq: 1656 }  },  {  model: 'AMD Ryzen 5 3550H with Radeon Vega Mobile Gfx ',  speed: 2096,  times: { user: 94734, nice: 0, sys: 97968, idle: 1049656, irq: 1468 }  },  {  model: 'AMD Ryzen 5 3550H with Radeon Vega Mobile Gfx ',  speed: 2096,  times: { user: 81906, nice: 0, sys: 47234, idle: 1113218, irq: 1312 }  },  {  model: 'AMD Ryzen 5 3550H with Radeon Vega Mobile Gfx ',  speed: 2096,  times: { user: 58578, nice: 0, sys: 38953, idle: 1144828, irq: 1046 }  },  {  model: 'AMD Ryzen 5 3550H with Radeon Vega Mobile Gfx ',  speed: 2096,  times: { user: 59406, nice: 0, sys: 34625, idle: 1148328, irq: 859 }  },  {  model: 'AMD Ryzen 5 3550H with Radeon Vega Mobile Gfx ',  speed: 2096,  times: { user: 55937, nice: 0, sys: 33125, idle: 1153296, irq: 843 }  },  {  model: 'AMD Ryzen 5 3550H with Radeon Vega Mobile Gfx ',  speed: 2096,  times: { user: 61718, nice: 0, sys: 31375, idle: 1149265, irq: 875 }  }  ]  Total Memory : 14901878784  Free Memory : 6341722112 |

* 1. **Path Module :**

//PATH Module :(We need a File to work with path Eg:f1.txt )

//NOTE : USE DOUBLE SLASHES in path.

const path = require('path');//import path Module.

//1.extname : To know the extension of a File in a Path:

let ext = path.extname('E:\\MCA\\COURSES\\NODE\_JS\\nodejs\_certification@scalar\\Module\_3\\f1.txt');

//2.basename : To know the extension of a File in a Path:

let base = path.basename('E:\\MCA\COURSES\\NODE\_JS\\nodejs\_certification@scalar\\Module\_3\\f1.txt');

console.log(ext);

console.log(base);

//3.To display the path of current file (this file)

console.log(\_\_filename);

//4.To display the directory of current file (this file)

console.log(\_\_dirname);

/\*

output :

.txt

f1.txt

e:\MCA\COURSES\NODE\_JS\nodejs\_certification@scalar\Module\_3\pathModule.js

e:\MCA\COURSES\NODE\_JS\nodejs\_certification@scalar\Module\_3

\*/

* 1. **FS Module with Files [readFileSync(),writeFileSync(),appendFileSync(),unlinkSync() ].**

//FS Module with files :(TO handle Files)

const fs = require('fs');//import fs.

//Create f1.txt ,f2.txt and f3.txt

//1.Reading a file [readFileSync()]:

let fileContent = fs.readFileSync('f1.txt');

console.log('Data of FIle 1 - >' + fileContent);//Use '+' to convert buffer to string data.

//2.Writing in a File [writeFileSync()],(Data inside will be overwritten) :

fs.writeFileSync('f2.txt','File 2 is Overwritten');//File 2 will be overwritten

//Even if f2.txt is not there it will create it.

console.log('File has been written');

//3.Append to a File[appendFileSync()],(Updating a File) :

fs.appendFileSync('f3.txt','Updating File 3');

console.log('File has been appended');

// 4.Delete a FIle [unlinkSync()] :

fs.unlinkSync('f2.txt');

console.log('File has been deleted.');

/\*OUTPUT:

PS E:\MCA\COURSES\NODE\_JS\nodejs\_certification@scalar\Module\_3> node fs.js

Data of FIle 1 - >Hi i am file 1

File has been written

File has been appended

\*/

* 1. **FS Module with Directories :**

// //FS Module With Directory:

const fs = require('fs');//import fs.

// 1.First Create Directory [mkdirSync()] :

fs.mkdirSync('myDirectory');

// //2.Check the content inside of a  Directory [readdirSync()]:

let folderPath = 'E:\\MCA\\COURSES\\NODE\_JS\\nodejs\_certification@scalar\\Module\_3\\myDirectory';

let folderContent = fs.readdirSync(folderPath);

console.log('Folder Content : ' , folderContent);

/\*OUTPUT :

Folder Content :  [ 'f1.txt', 'f3.txt' ]

\*/

//3.Check particular directory exists or not [existSync()]:

let doesExist = fs.existsSync('myDirectory');

console.log(doesExist);//true

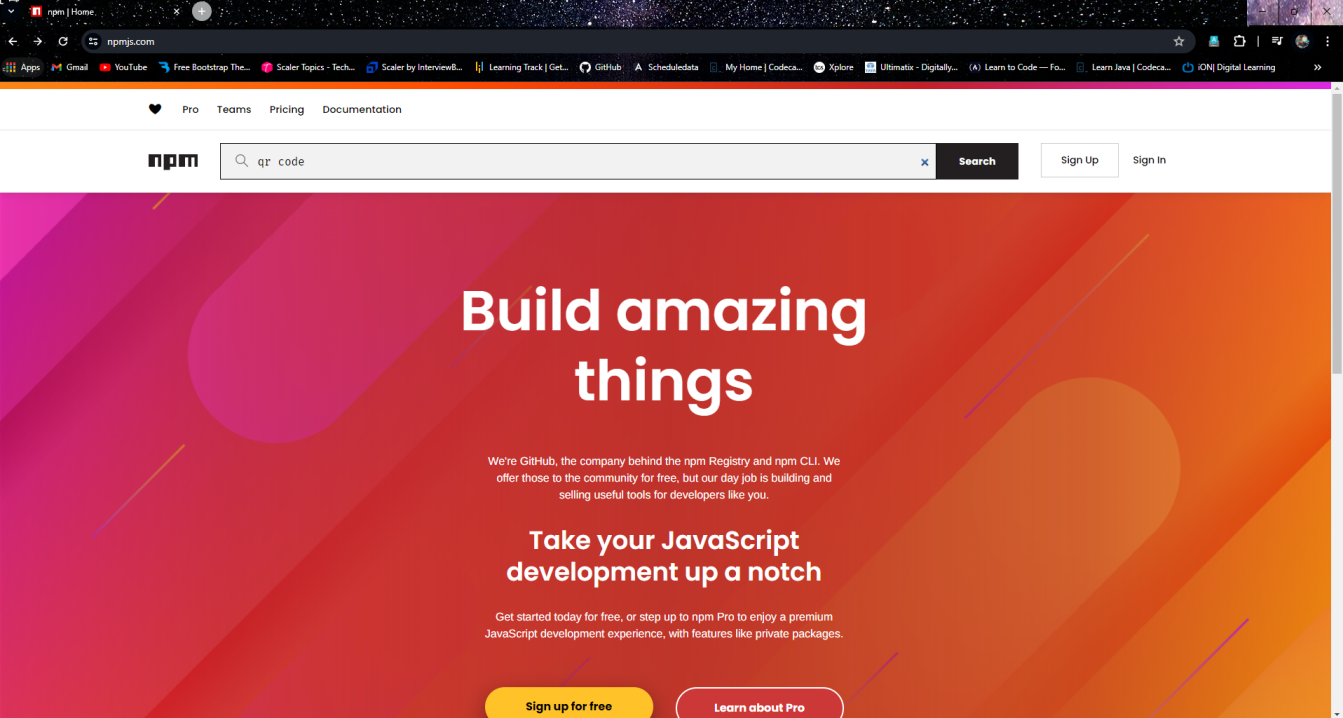
//4.Remove Directory []:

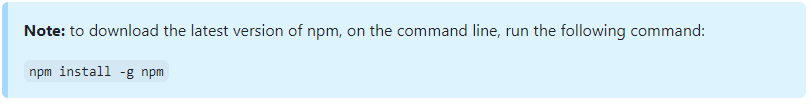
//Before that empty the directory.

fs.rmdirSync('myDirectory');

console.log('Directory Has been Deleted');

1. **Node Package Manager(NPM):[Third party packages that we can use in JS]**
   1. **Introduction to NPM :(We can use npm install different packages)**





* 1. **How to Install and use an NPM Package :**

1. **Create a folder and open it in terminal and run ‘npm init’:**

npm init

PS E:\MCA\COURSES\NODE\_JS\nodejs\_certification@scalar\Module\_4\npm\_package> npm init

This utility will walk you through creating a package.json file.

It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields

and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and

save it as a dependency in the package.json file.

Press ^C at any time to quit.

package name: (npm\_package)

version: (1.0.0)

description: it is my package

entry point: (index.js)

test command:

git repository:

keywords:

author:

license: (ISC)

About to write to E:\MCA\COURSES\NODE\_JS\nodejs\_certification@scalar\Module\_4\npm\_package\package.json:

{

  "name": "npm\_package",

  "version": "1.0.0",

  "description": "it is my package",

  "main": "index.js",

  "scripts": {

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

  },

  "author": "",

  "license": "ISC"

}

Is this OK? (yes)

1. **“package.json” file will be created inside that folder :**

{

  "name": "npm\_package",

  "version": "1.0.0",

  "description": "it is my package",

  "main": "index.js",

  "scripts": {

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

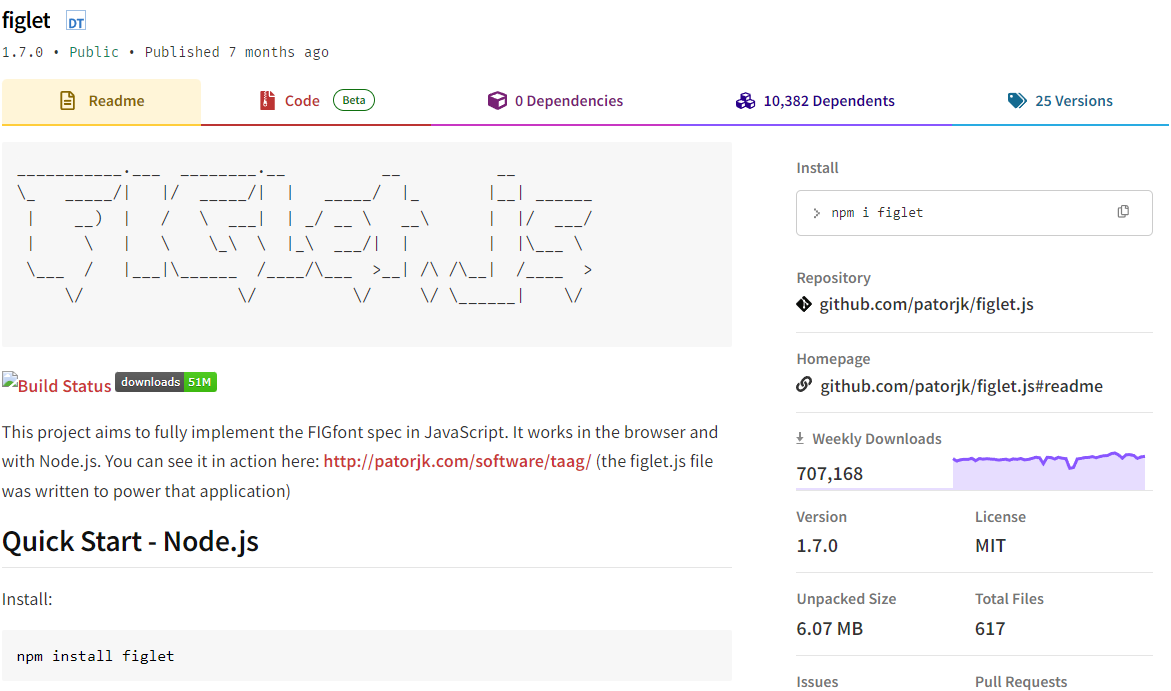
  },

  "author": "",

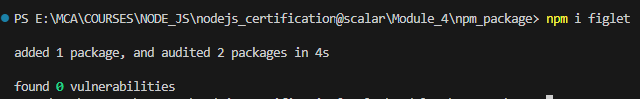
  "license": "ISC"

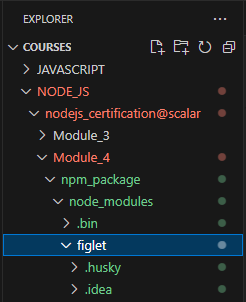
}

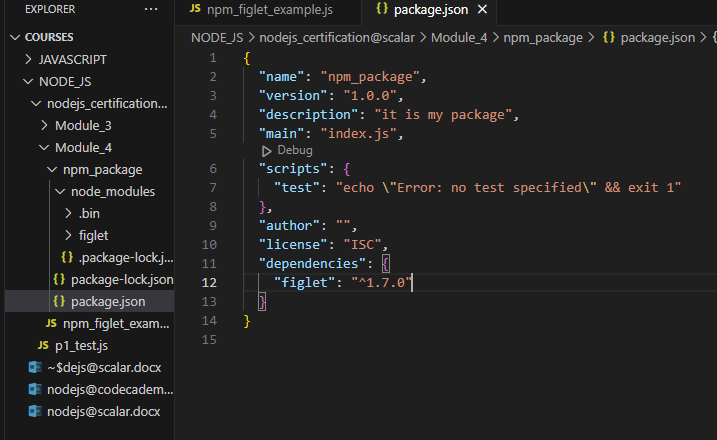
1. **Example : Installing ‘figlet’ package :**
2. **Goto npm site and type figlet :**



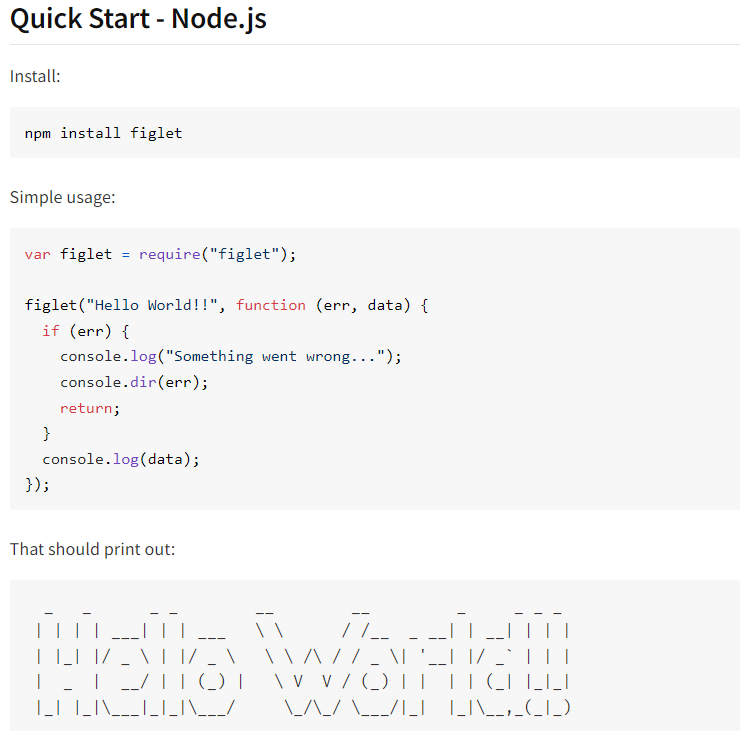
1. **Install figlet :**







1. **Add example code in a new file ‘npm\_figlet\_example.js’ :**



const figlet = require('figlet'); //figlet package which we installed is  imported.

figlet("Hello World!!", function (err, data) {

  if (err) {

    console.log("Something went wrong...");

    console.dir(err);

    return;

  }

  console.log(data);

});

PS E:\MCA\COURSES\NODE\_JS\nodejs\_certification@scalar\Module\_4\npm\_package> node .\npm\_figlet\_example.js

  \_   \_      \_ \_        \_\_        \_\_         \_     \_ \_ \_

 | | | | \_\_\_| | | \_\_\_   \ \      / /\_\_  \_ \_\_| | \_\_| | | |

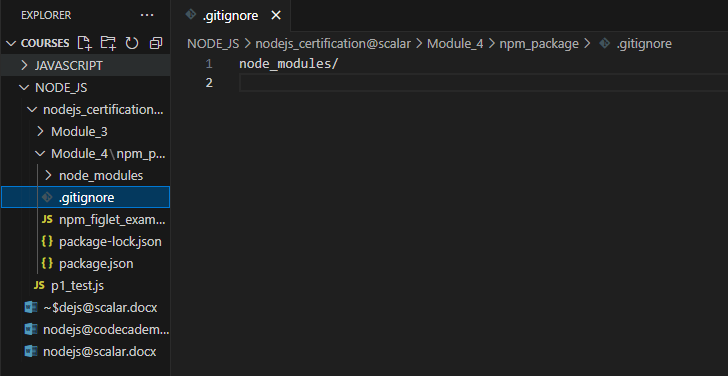
 | |\_| |/ \_ \ | |/ \_ \   \ \ /\ / / \_ \| '\_\_| |/ \_` | | |

 |  \_  |  \_\_/ | | (\_) |   \ V  V / (\_) | |  | | (\_| |\_|\_|

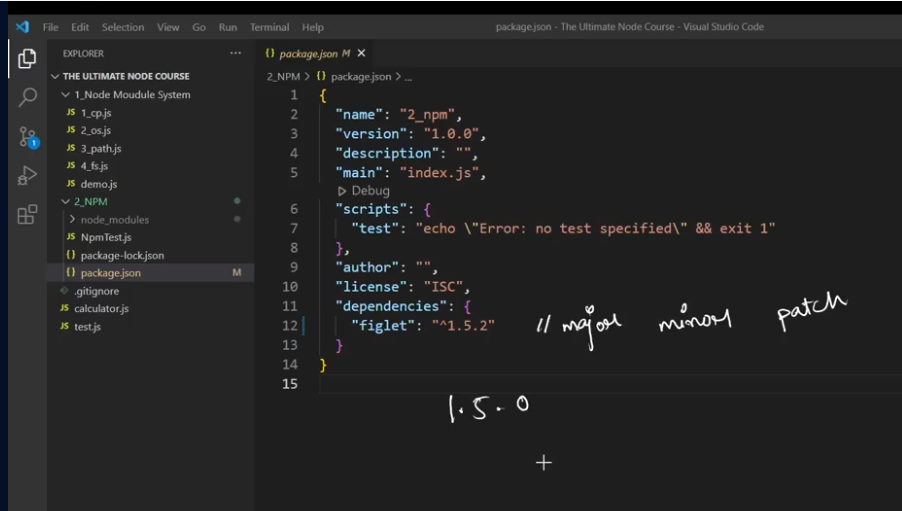
 |\_| |\_|\\_\_\_|\_|\_|\\_\_\_/     \\_/\\_/ \\_\_\_/|\_|  |\_|\\_\_,\_(\_|\_)

PS E:\MCA\COURSES\NODE\_JS\nodejs\_certification@scalar\Module\_4\npm\_package>

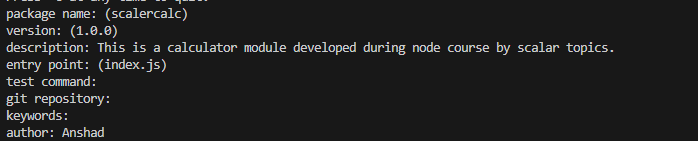
* 1. **All about ‘.gitignore ‘ :**



* 1. **Semantic Versioning :(major , minor , patching)**



* 1. **Publishing your own NPM package :**
     1. **First create a folder ‘scalarCalc’ .**
     2. **Then install npm : ‘npm init’.**

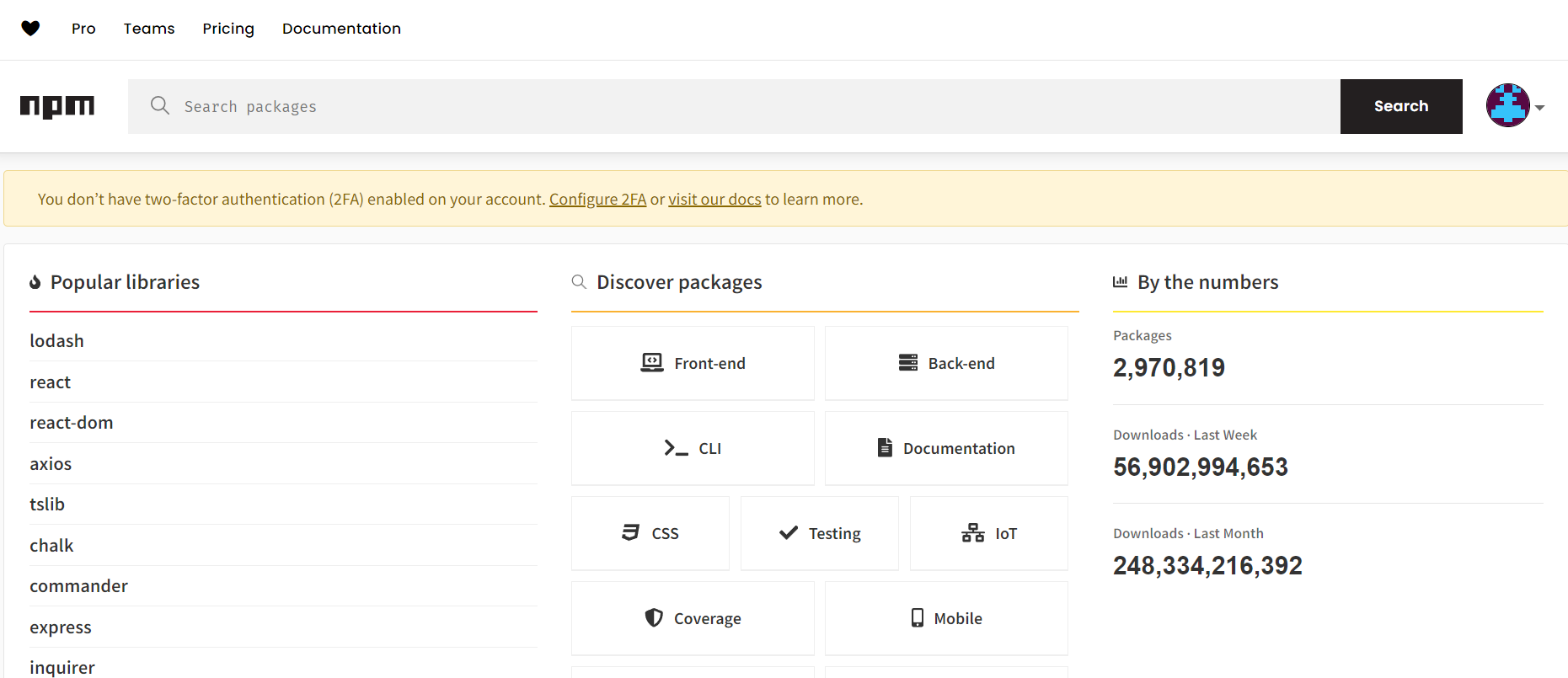


* + 1. **Now create an Account in npm .(Adding a user).It will take us into a page to create a user.**

npm adduser

* + 1. **Now Login to that .**

npm login



* + 1. **Create a file ‘index.js’ inside ‘scalarCalc’ :**

//CALCULATOR :

function add(a , b ){

    console.log( a + b );

}

function sub(a , b ){

    console.log( a - b );

}

function mul(a , b ){

    console.log( a \* b );

}

function div(a , b ){

    console.log( a / b );

}

//Exporting functions:

module.exports = {

    addition : add ,

    subtraction : sub,

    multiplication : mul,

    division : div

}

* + 1. **Now publish it :**

npm publish

**Now Package is published to npm.**

* + 1. **Now GOTO npm site and type your package name : ‘scalarCalc’ .**
    2. **Where you can see the published npm.**
    3. **Now You can use it by :**

1. **Create a test folder.**
2. **“npm init --yes”.**
3. **“npm i scalarCalc”.**
4. **Now we can use it.by importing**

const calc = require('scalarCalc');

1. **Asd**
2. **Asd**
3. **Asd**
4. **Asd**
5. **asda**