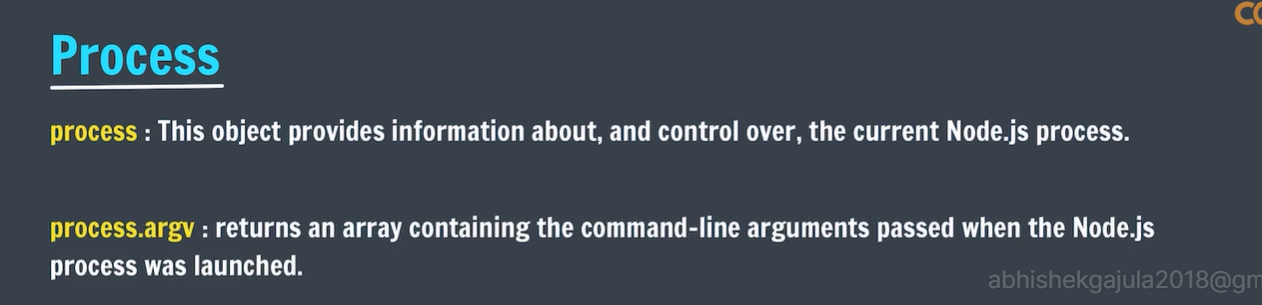
* Node.js

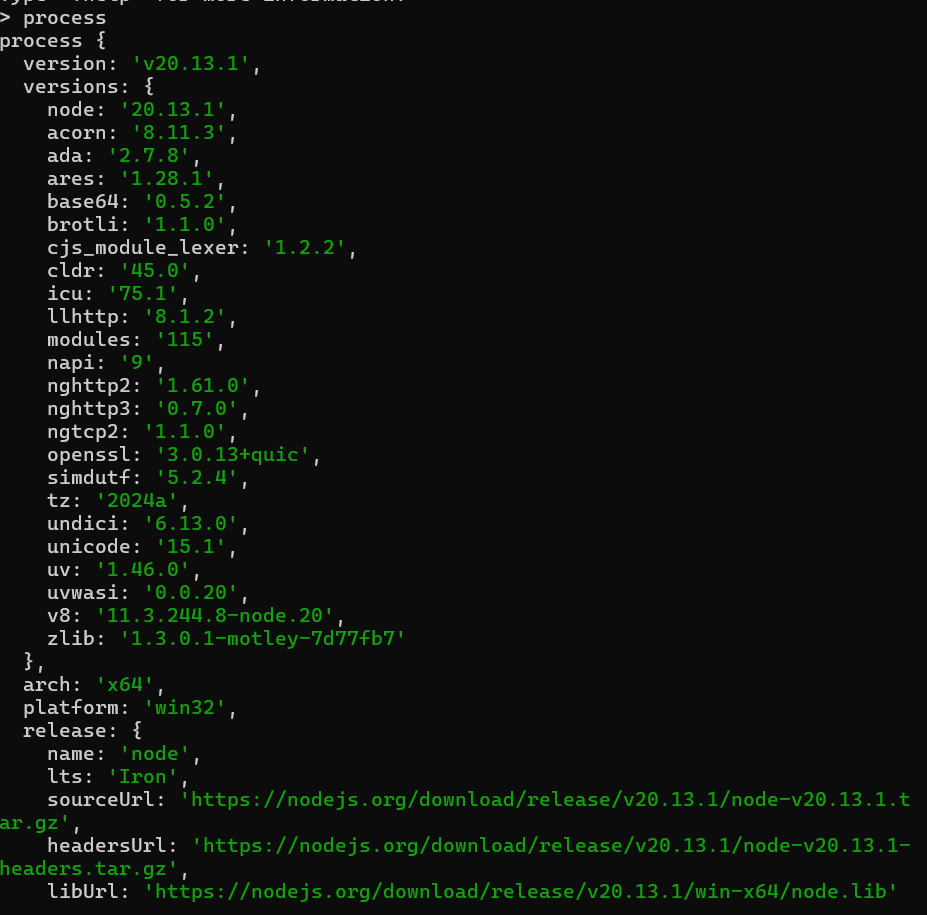
## What is node js:

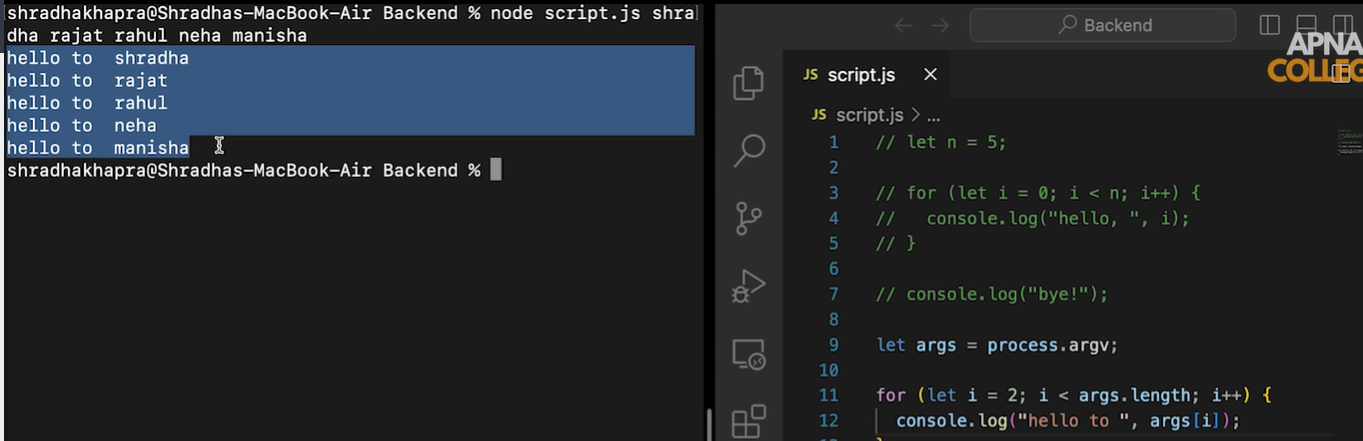
**Node.js** (Node js) is an **open-source** **and** **cross-platform**JavaScriptruntime environment. It runs on **Chrome’s V8 JavaScript engine**. It allows developers to run JavaScript code on the server. Node.js enables developers to get into the **server-side world.**



* “touch” is used to create file in cmd.





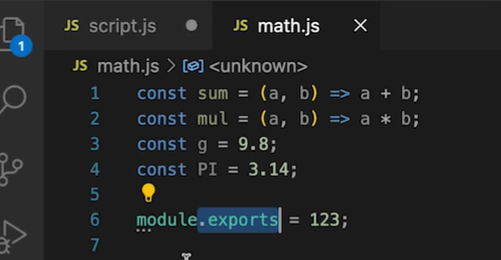
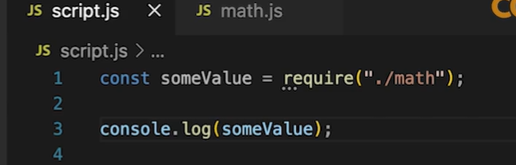


* First two arguments will be the node bin path and current working directory path.

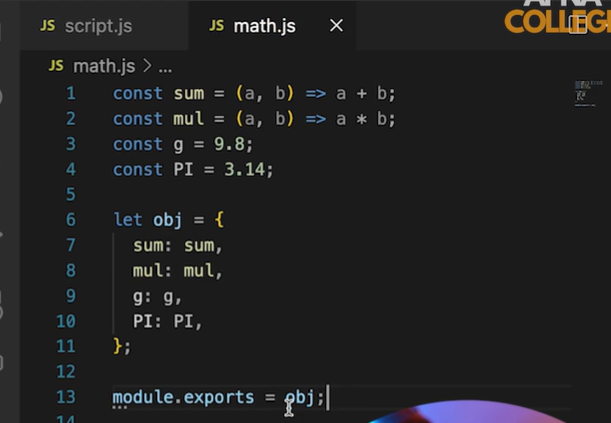
# Who to share data between js files:

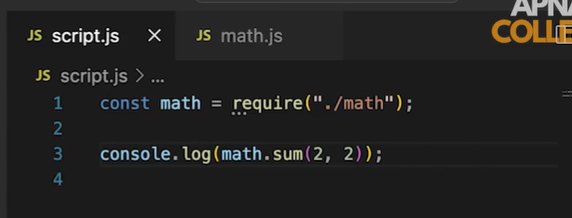


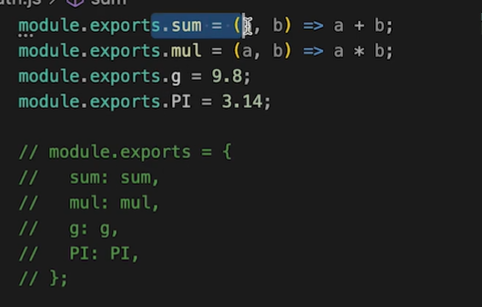
* You have function and some constants which are more often can used in other programs then this features will come into picture.



* If no data is there to export then require object will return empty object.

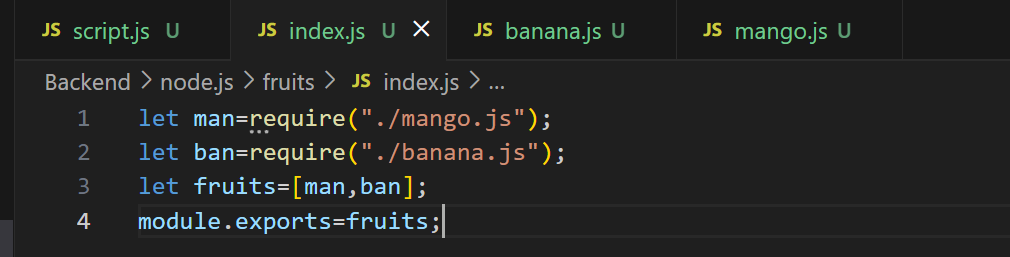






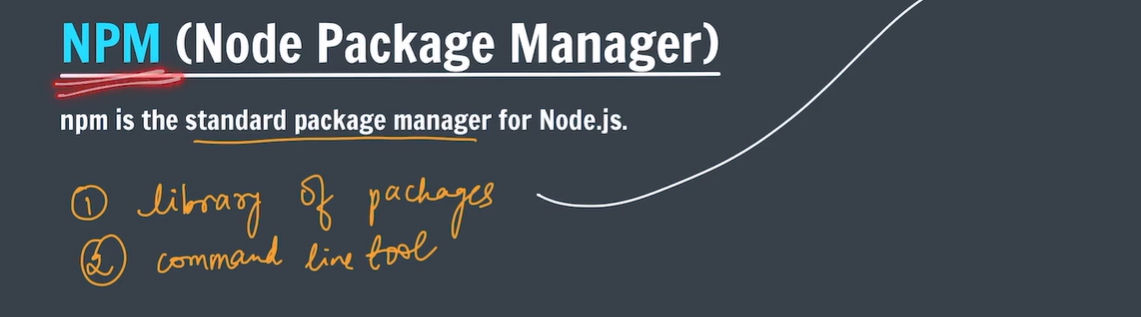
# Sharing the data between the different directories:

* Sharing among directories is same as sharing among files but the sharing directory should have an entry point which is index.js .
* In index.js we should collect all the data belongs to that folder and export form the file itself



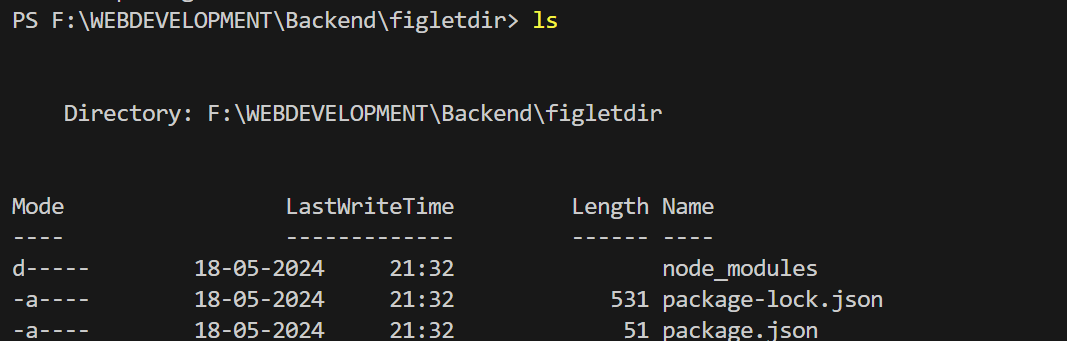
* And file that is collecting the data from the other folder should use require function with that folder function.

# Npm(node package manager):

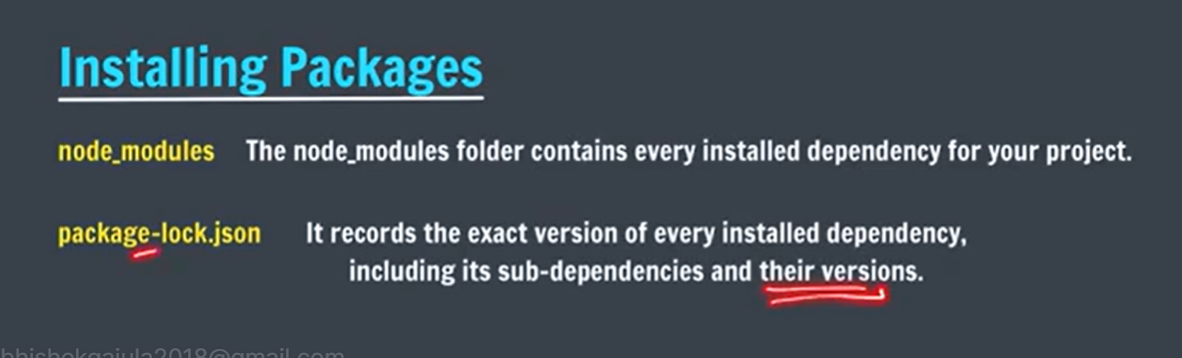


* Npm is a library of packages which manages the packages like express and react .
* It also provide command line tool which is use to download the packages to the developers.





* If we download figlet the it contains above three above files by default.



* Note : while using those packages we must and should be in the same directory where those packages are installed.

Install:

npm install figlet

Simple usage:

var figlet = require("figlet");

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

if (err) {

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

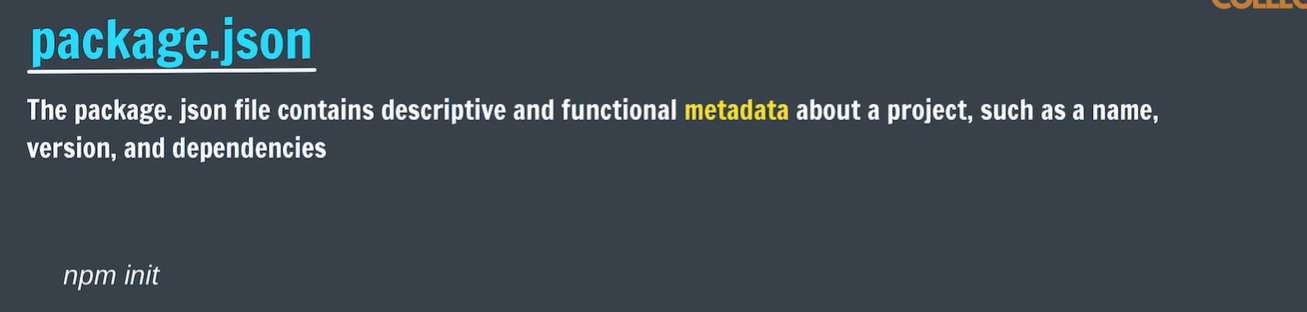
console.dir(err);

return;

}

console.log(data);

});



* Suppose if you want to send a project which is made by using some packages.you no need to send node modules you have to send only package.json in that you have all dependencies and functionality ,versions etc related to package used.
* For suppose if lost node module file we no need to worry we can re download with package.json file.
* Eg: delete the nodemodules and type “npm install” then we can easily re download the node modules.

# How to create package.json in our own project.

* Create a new directory .
* Go to that directory
* And type the command “npm init”

PS F:\WEBDEVELOPMENT\Backend\temp> 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: (temp)

version: (1.0.0)

description: how to use node in our new project

entry point: (index.js)

test command:

git repository:

keywords:

author: abhishek

license: (ISC)

About to write to F:\WEBDEVELOPMENT\Backend\temp\package.json:

{

"name": "temp",

"version": "1.0.0",

"description": "how to use node in our new project",

"main": "index.js",

"scripts": {

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

},

"author": "abhishek",

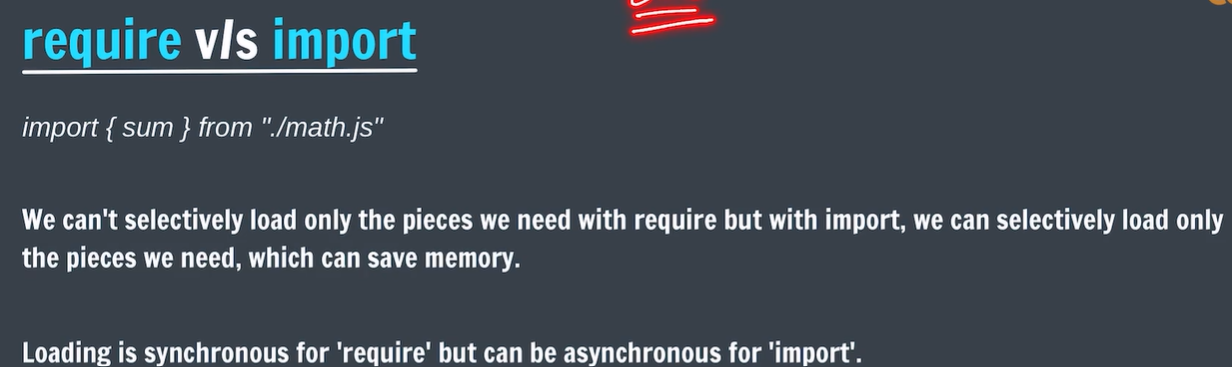
"license": "ISC"

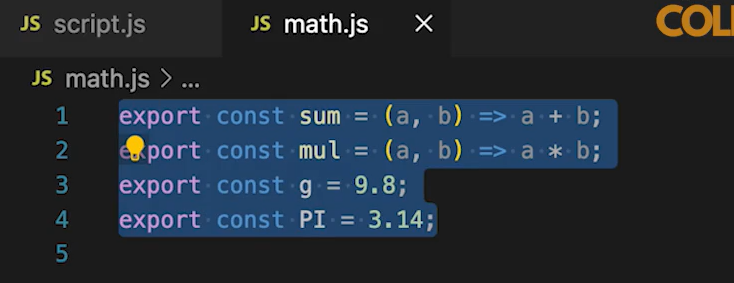
}

Is this OK? (yes)

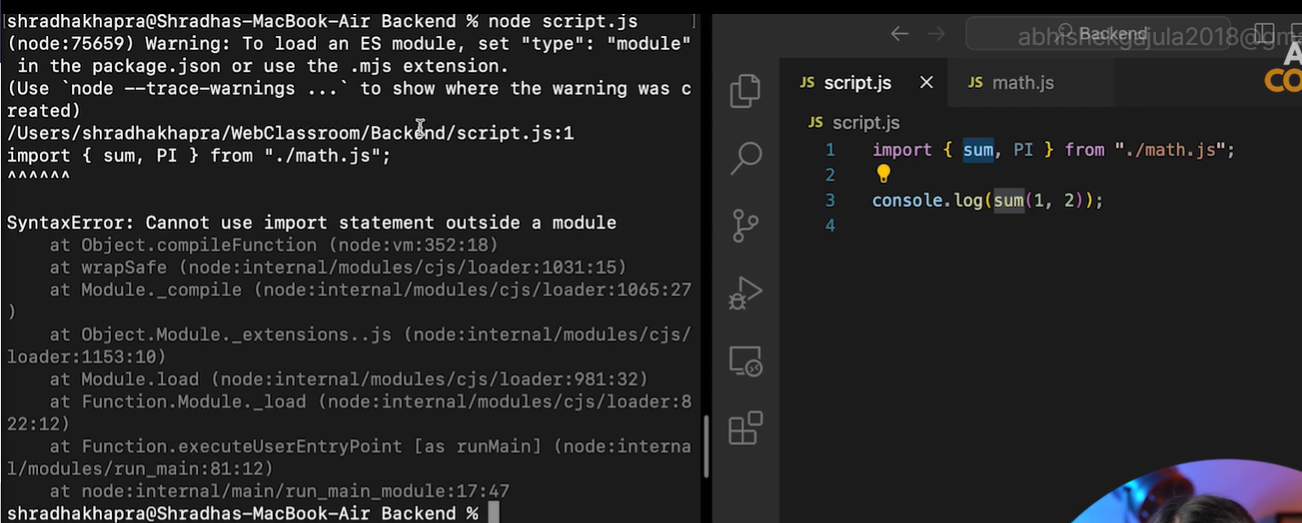
PS F:\WEBDEVELOPMENT\Backend\temp>



* First is used to install the packages globally.
* Second command is used link the package and current working directory.
* After linking the package to our current working directory we can use that package.
* Note: you have to use either require or import but both is not recommended.
* Eg. Write the keyword ‘export’ before the statements which you want use in other folders.

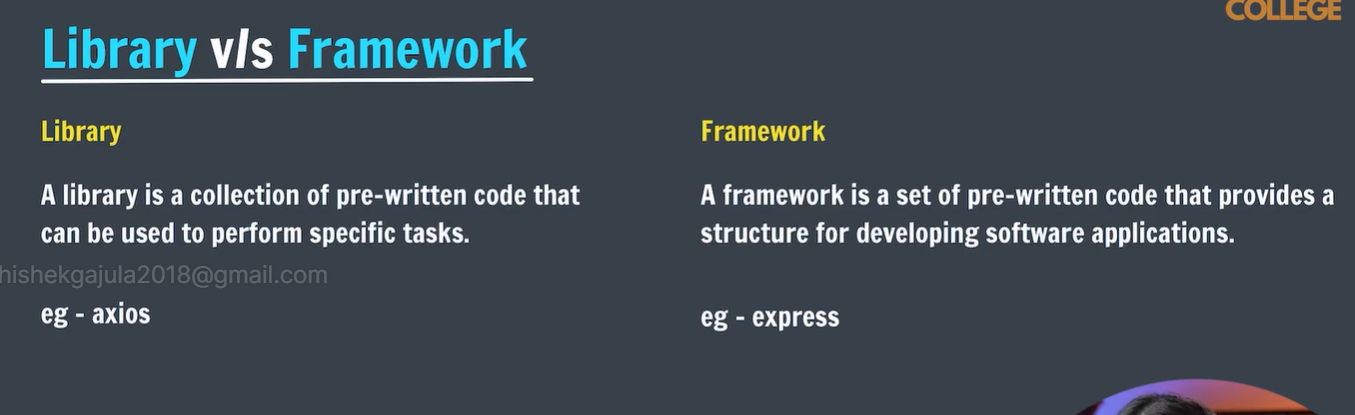


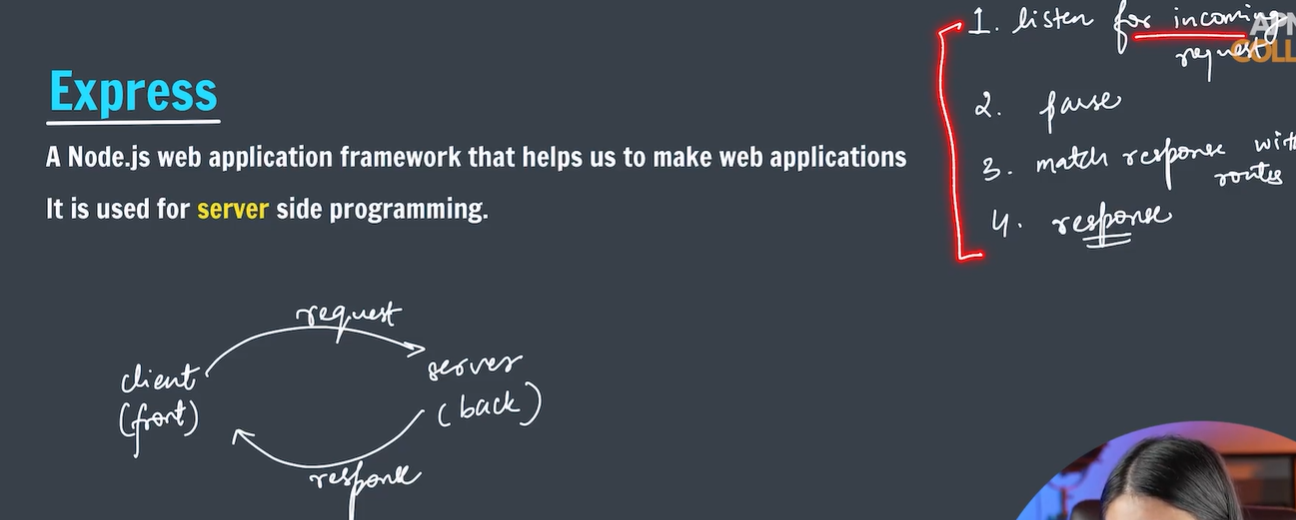
* Write the import in the file in which you want use other module features.



* It will show an error that in parent directory there should be a package.json file and in that a key value pair should contain “type”:”module”:

# Express js:





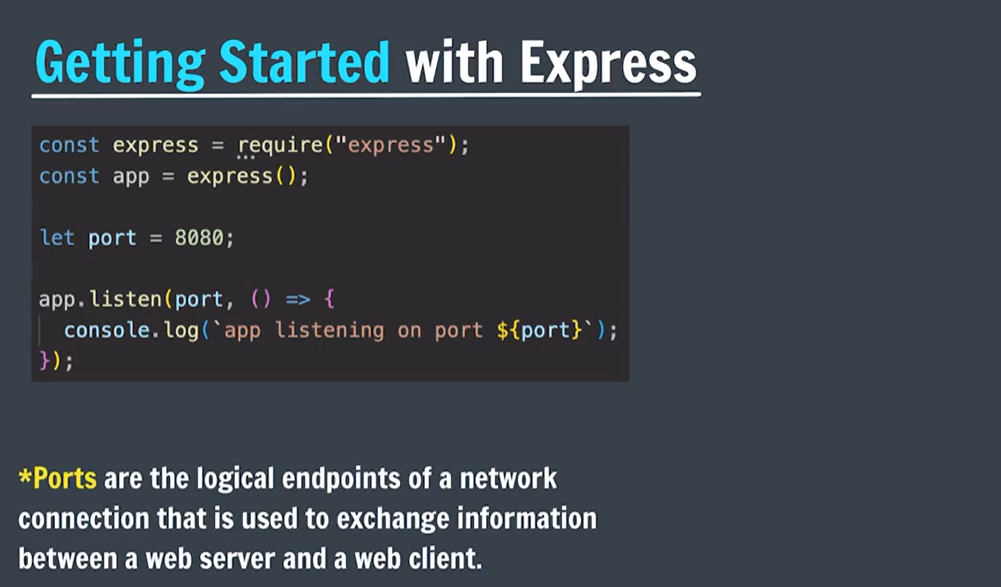
* It will listen for incoming request.
* Parse the incoming request.
* Match the response with routes.

Routes means the key words which is written after the domain and slash which is used to navigate to particular page of the website.

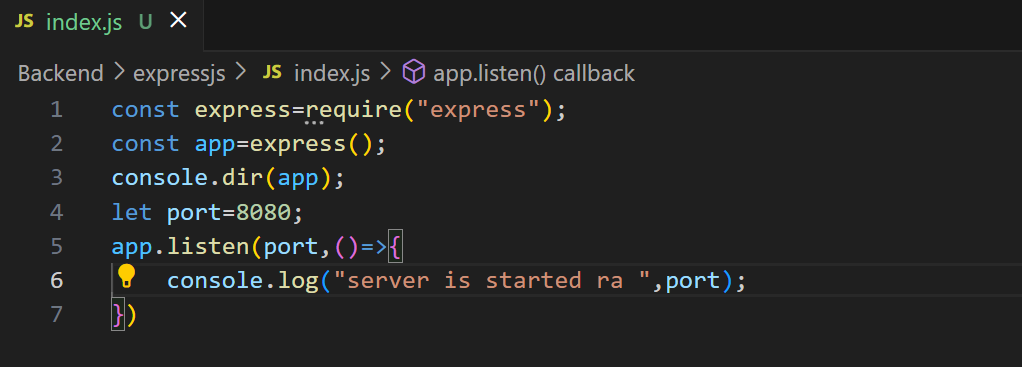
* Response.

# Installation of express:

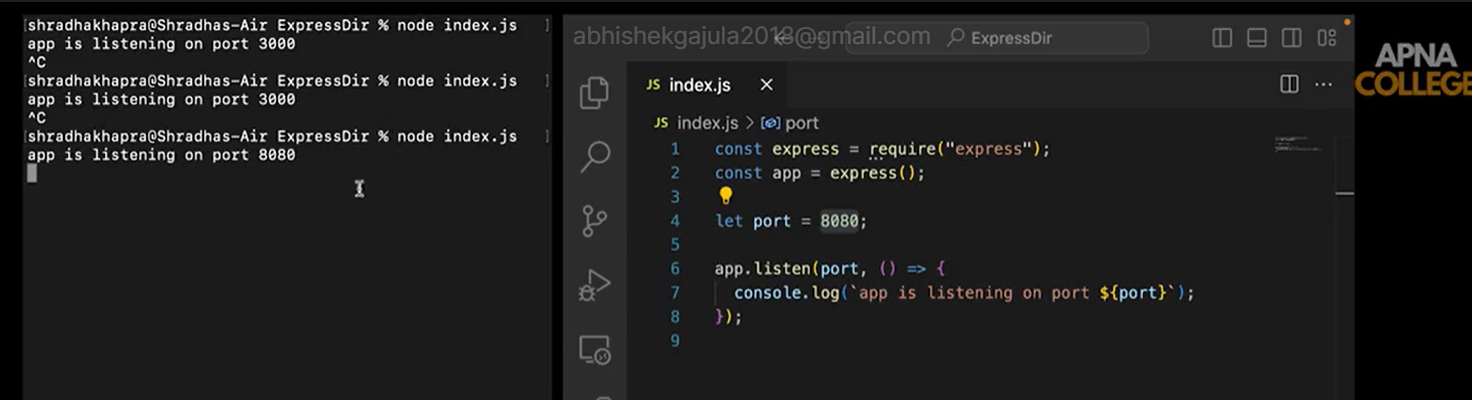
* First create a dir and navigate to that dir and type “npm install”.
* Becoz it creates a package.json which provides some functionalities such as using of import statement and information of package dependencies. And it also add basic information about your project.



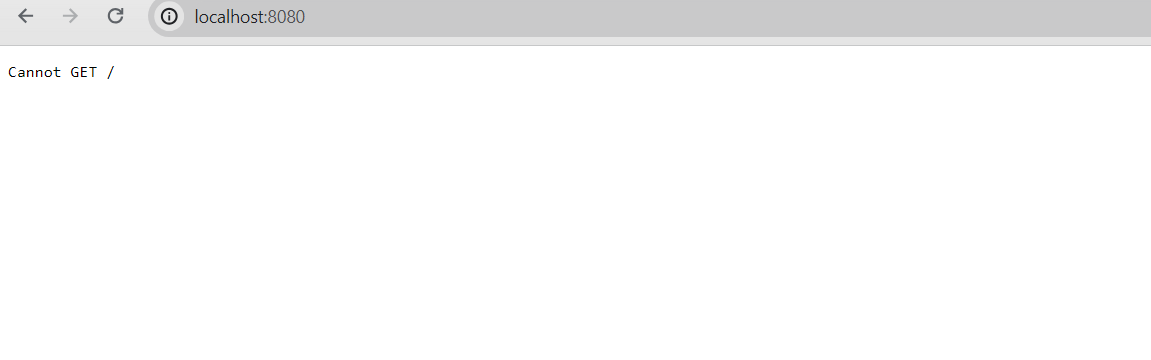
* Getting started with the server:



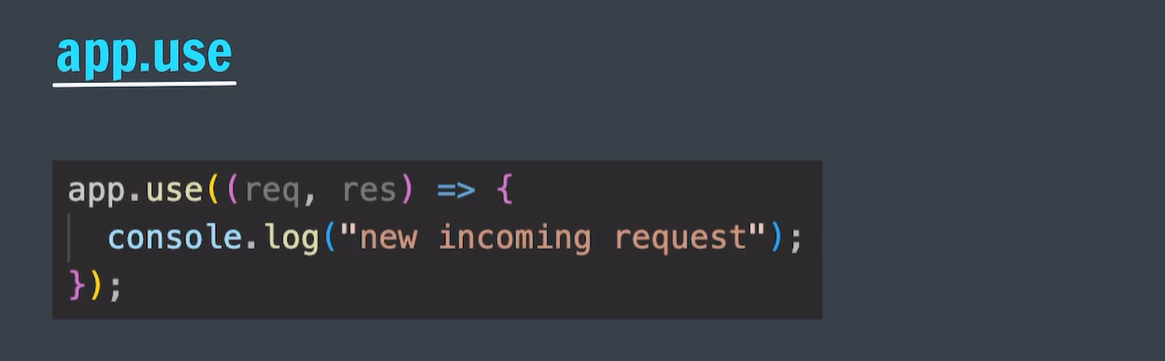
* Write the above program and name it as index.js give a port number (some port number are predefined for specific purpose ).
* Listen is the method of express object which always listen to the requests.



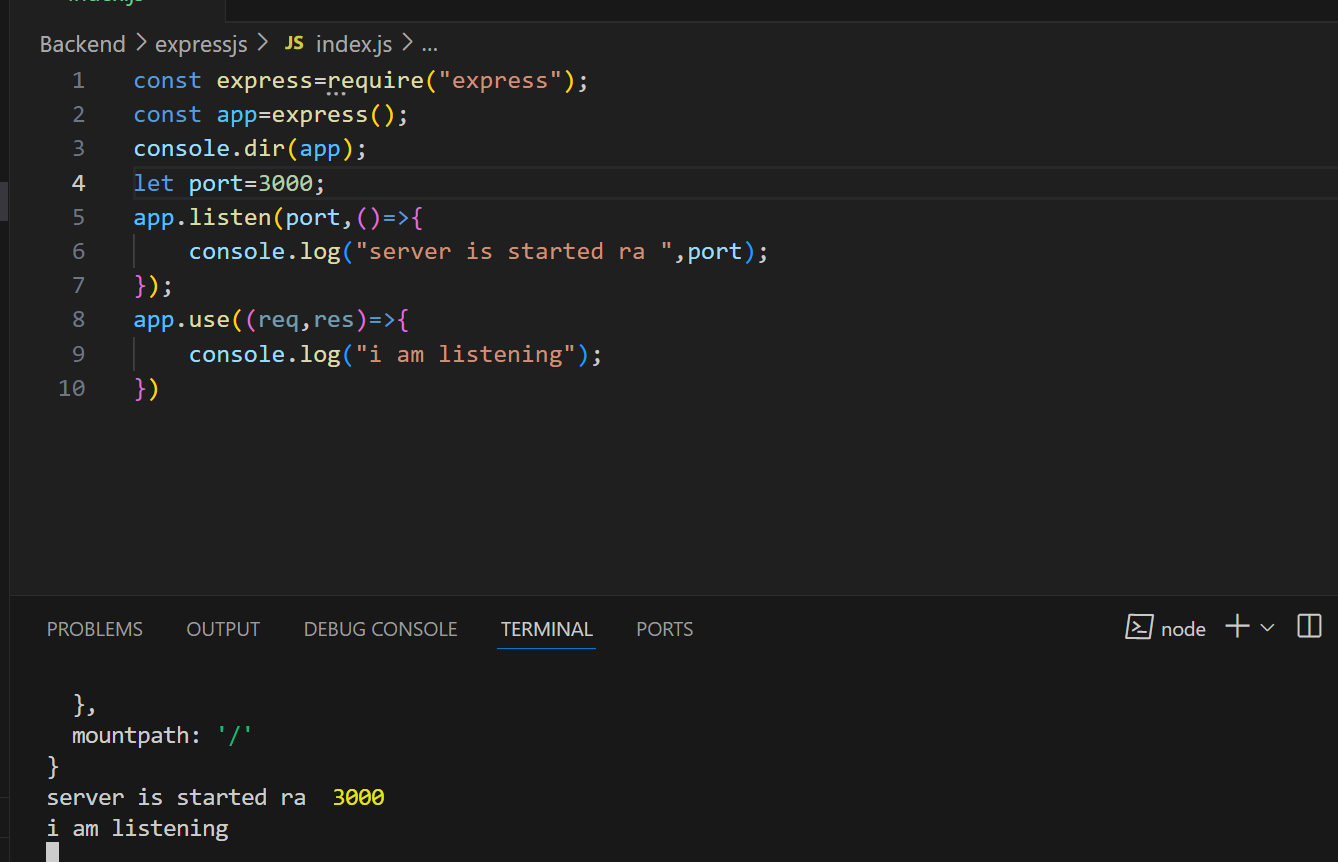
* Once if start the program by typing node index.js it will always listen the request until we stop it.
* To stop the server type”ctrl+c”
* We you want check the server working or not type”localhost:<portnumber>”



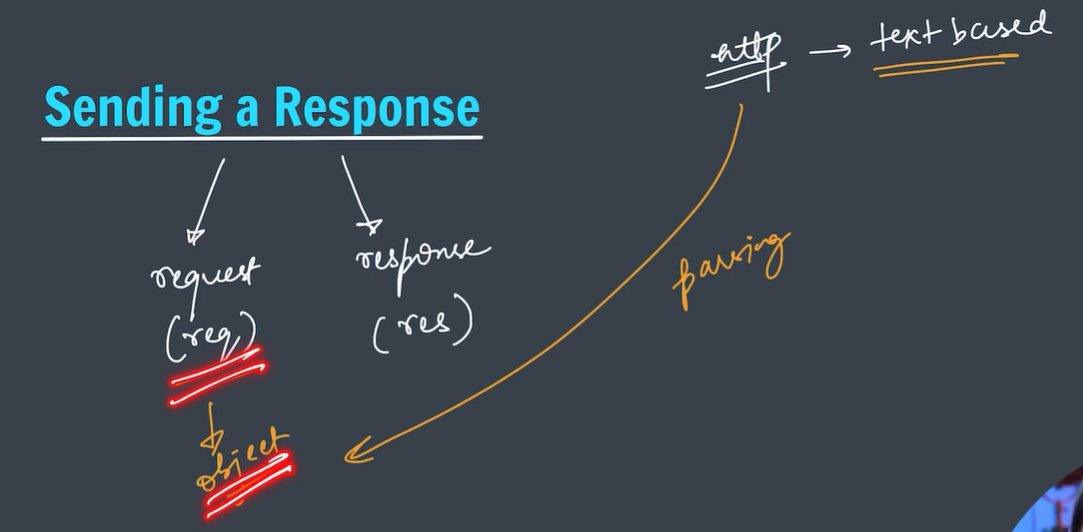
* We stop execution of index.js by “ctrl+c” it will stop listening and this case will happen even if you type wring port number.



* It is used send the response to any type of request.
* You can check that In hopscotch but it doesn’t work with local api so you follow the given link to do that [**https://github.com/hoppscotch/hoppscotch/discussions/2051**](https://github.com/hoppscotch/hoppscotch/discussions/2051)
* **The above function has two default parameters req and res.**
* After downloading the extension refresh the page and restart the server.



* I am listening will be printed when you send the request to the browser on <http://localhost:3000>



* Basically http request are text based why because they should not depend on server architecture.
* Express will convert that request to js object which is one of the main function of js called parsing.

### res.send([body])

Sends the HTTP response.

The body parameter can be a Buffer object, a String, an object, Boolean, or an Array. For example:

res.send(Buffer.from('whoop'))

res.send({ some: 'json' })

res.send('<p>some html</p>')

res.status(404).send('Sorry, we cannot find that!')

res.status(500).send({ error: 'something blew up' })

This method performs many useful tasks for simple non-streaming responses: For example, it automatically assigns the Content-Length HTTP response header field (unless previously defined) and provides automatic HEAD and HTTP cache freshness support.

When the parameter is a Buffer object, the method sets the Content-Type response header field to “application/octet-stream”, unless previously defined as shown below:

res.set('Content-Type', 'text/html')

res.send(Buffer.from('<p>some html</p>'))

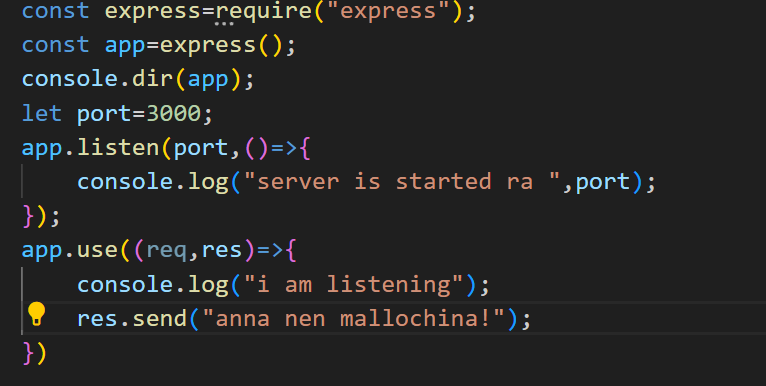
When the parameter is a String, the method sets the Content-Type to “text/html”:

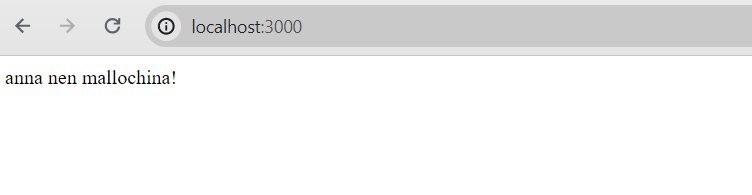
res.send('<p>some html</p>')

When the parameter is an Array or Object, Express responds with the JSON representation:

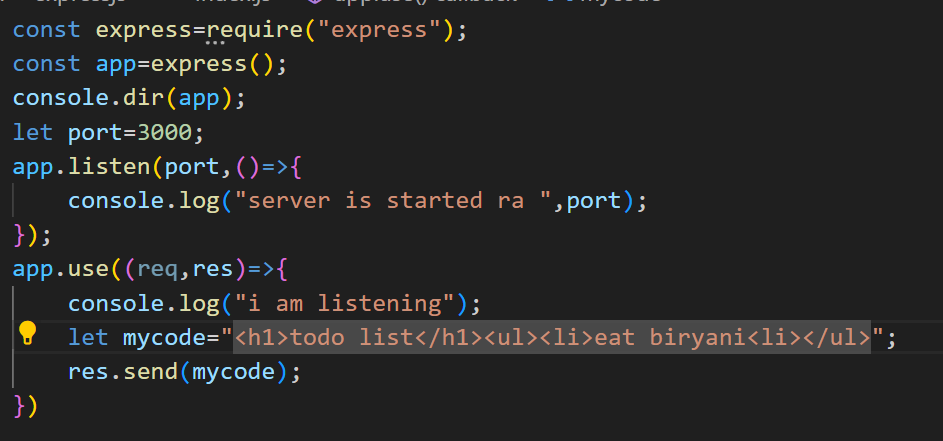
res.send({ user: 'tobi' })

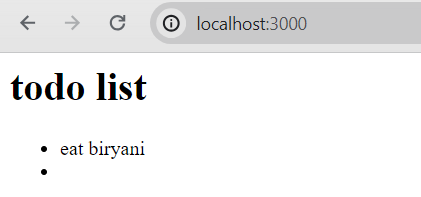
res.send([1, 2, 3])

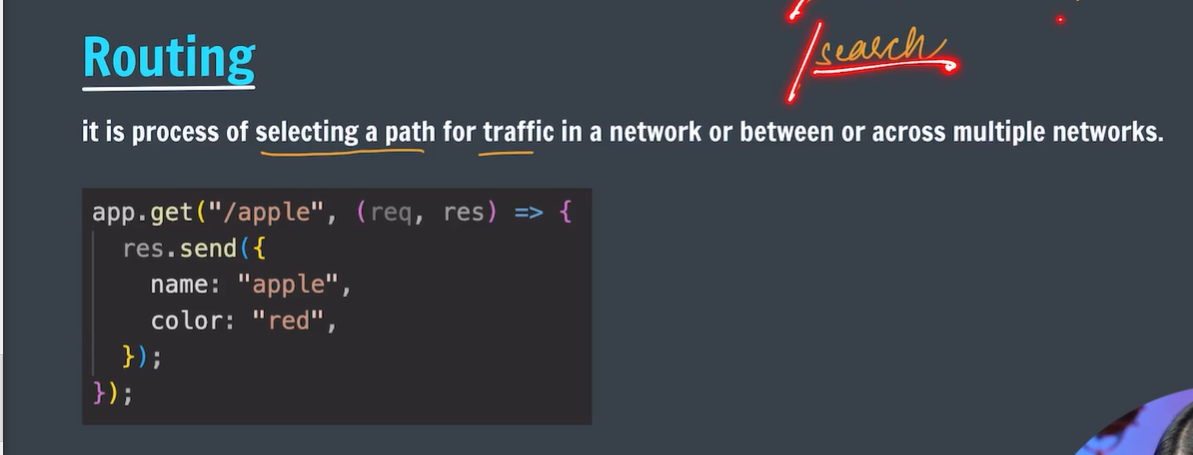




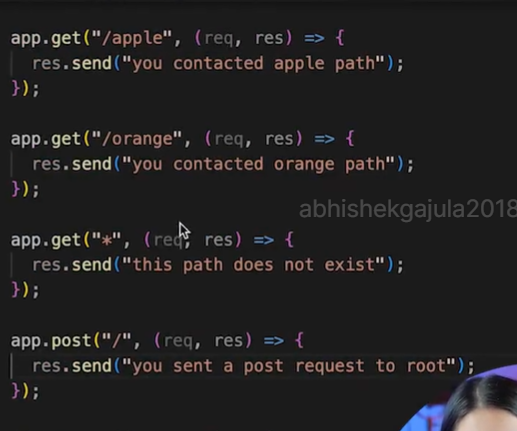
* Send method automatically converts it into json format.

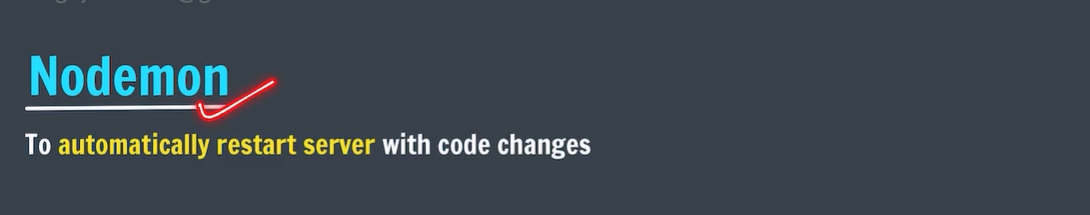






* App.use will listen all type of request even if there are different routes are there.
* It will respond to only get type of requests.





* It is good to save it in global scope.

PS F:\WEBDEVELOPMENT\Backend\expressjs> npm install -g nodemon

added 29 packages in 8s

4 packages are looking for funding

run `npm fund` for details

* Now you can check by command “nodemon -v”

# Error alert:

PS F:\WEBDEVELOPMENT\Backend\expressjs> nodemon -v nodemon : File C:\Users\user\AppData\Roaming\npm\nodemon.ps1 cannot be loaded because running scripts is disabled on this system. For more information, see about\_Execution\_Policies at https:/go.microsoft.com/fwlink/?LinkID=135170. At line:1 char:1 + nodemon -v + ~~~~~~~ + CategoryInfo : SecurityError: (:) [], PSSecurityException + FullyQualifiedErrorId : UnauthorizedAccess

The error message you are encountering is due to the execution policy settings in PowerShell, which prevent the running of scripts. To resolve this, you need to change the execution policy. Here are the steps to do that:

1. **Open PowerShell as Administrator:**
   * Search for "PowerShell" in the Start menu.
   * Right-click on "Windows PowerShell" and select "Run as administrator".
2. **Change the Execution Policy:**
   * In the PowerShell window, enter the following command to set the execution policy to allow scripts to run:

powershell

Copy code

Set-ExecutionPolicy RemoteSigned -Scope CurrentUser

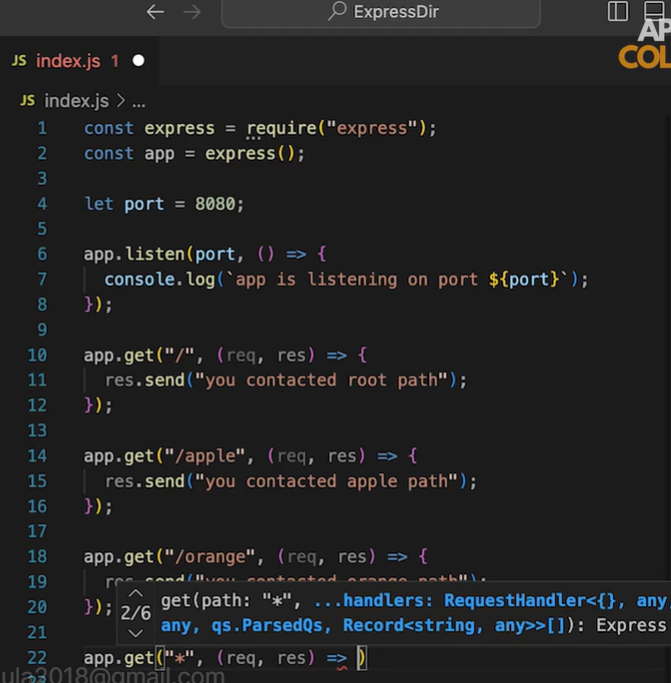
* + Press **Enter**.

1. **Confirm the Change:**
   * You might be prompted to confirm the change. Type **Y** and press **Enter**.
2. **Verify the Change:**
   * You can verify that the policy has been changed by running:

powershell

Copy code

Get-

* 
* The responding of get method is restricted to certain requests only .
* The last get method Indicates it responsible for all the request other than specified above.
* Unable to declare get method for every type route.to overcome this we use path parameters.

