NODE JS

1.What is Node JS ?

- JS Runtime Environment
- Opensource, Cross-platform , and Runtime environment for executing JS code outside a browser
- Before node JS the JS code run on the environment of the browser.
- Node JS is not a programming language and not a framework

2.Features of Node JS

- Cross-platform compatibility
- Asynchronous (Non-blocking)
- Uses JS
- Fast data streaming
- Event driven

ADVANTAGES

FRONT-END BACKEND
HIGH PERFOMANCE
SCALABLE

Cross-Platform : It may be used on variety of systems , including windows , linux ,unix and mobile devices

Asynchronous : A server built with node JS never wait for data from an API

Fast data streaming : When data is transmitted in multiple stream processing them take a long time .Node JS process data at very rate

Event driven : It means as soon as node starts its server simply initiates its variables declare functions and then simply wait for event to occur.

3.Creating and Running basic Node JS Application

- Import required modules
- Create server
- Read request and return responses

Package.json
it is a blueprint or manifest of ou
project

It contails : Project info
Dependecies
Scripts

1.Import required modules

Load Node modules using required directive

Eg : var http = require(“http”)

2.Create server

Create a server to listen to the client request.

createServer() method is used for server creation

Then bind the server to port 8080 using the listen method associated with the server instance

http.createServer().listen(8080);

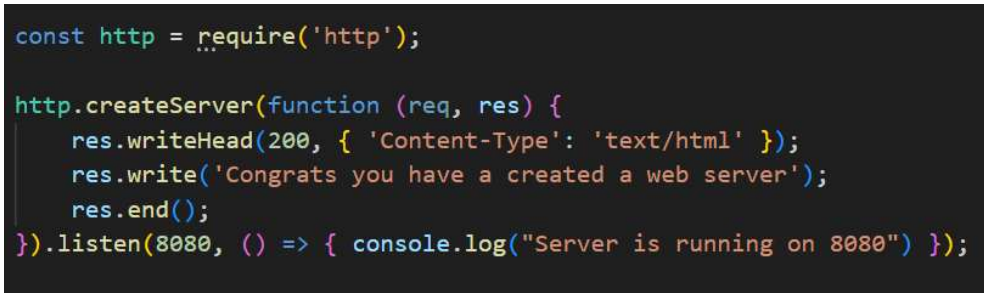
3.Read request and return responses

A function with request and response parameter is used to read client request and return responses.

http.createServer(function(req,res)=>{

....

}).listen(8080);



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

http.createServer(function (req, res) {
    res.writeHead(200, { 'Content-Type': 'text/html' });
    res.write('Congrats you have a created a web server');
    res.end();
}).listen(8080, () => { console.log("Server is running on 8080") });
```

4 . What is module in Node JS ?

A module is a set of JavaScript functions and objects that can be used in an application.

Modules are similar to JavaScript libraries and are the building blocks of Node.js applications.

5 . What are the different types of modules in Node Js ?

1. Core Module / In-built Module
2. Local Module

3. Third party Module / NPM Module

Core Module :

Node.js has many built-in modules that are part of the platform and come with **Node.js** installation.

These modules can be loaded into the program by using the **required** function.

Syntax : const module = require('module_name');

Examples : http, fs, url, path, os

Local Module :

Unlike built-in and external modules, local modules are created locally in your Node.js application.

Create our own modules and we can use it in our project

Third party Module :

Third-party modules are modules that are available online using the Node Package Manager(NPM).

Some of the popular third-party modules are Mongoose, express, angular, and React.

Example :

```
npm install express

npm install mongoose
```

6 . Node.js File System (FS)

It is used to handle file operations like creating , reading, deleting etc.

Node js provides an inbuilt module called FS

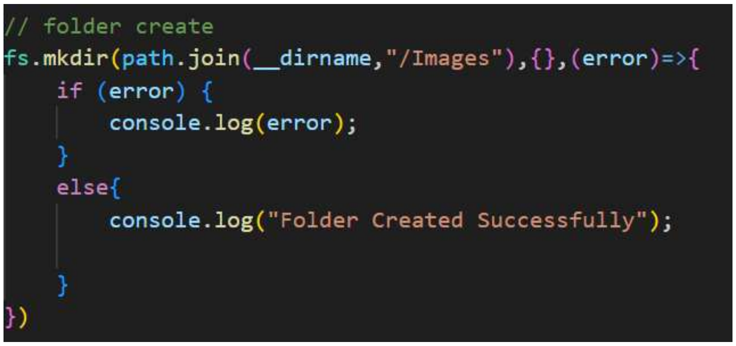
To use this module we need to require FS at first .

var fs = require('fs')

fs.mkdir()

It is used to create a directory.

Syntax : fs.mkdir(path, mode, callback)

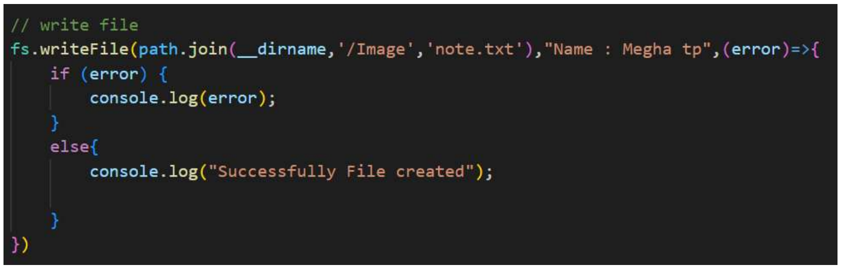


```
// folder create
fs.mkdir(path.join(__dirname,"/Images"),{},(error)=>{
    if (error) {
        console.log(error);
    }
    else{
        console.log("Folder Created Successfully");

    }
})
```

fs.writeFile()

It is used to asynchronously write the specified data to a file . By default the file would be replaced if it exists

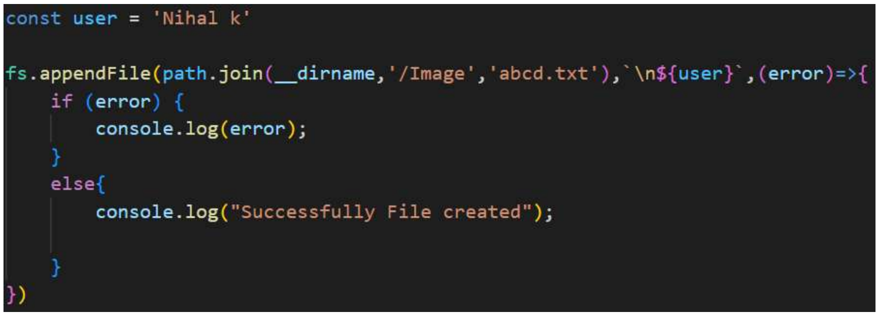


```
// write file
fs.writeFile(path.join(__dirname,'/Image','note.txt'),"Name : Megha tp",(error)=>{
    if (error) {
        console.log(error);
    }
    else{
        console.log("Successfully File created");

    }
})
```

fs.appendFile()

It is used to asynchronously append the given data to a file .



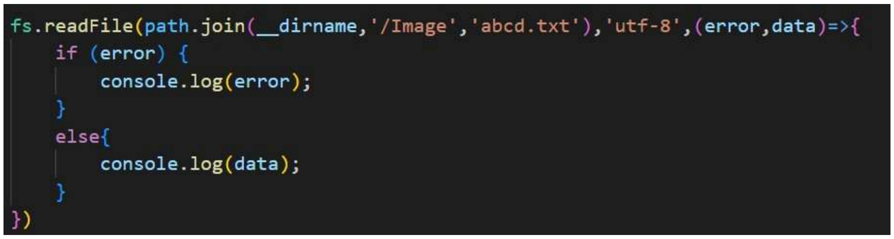
```
const user = 'Nihal k'

fs.appendFile(path.join(__dirname,'/Image','abcd.txt'),`\n${user}`,(error)=>{
    if (error) {
        console.log(error);
    }
    else{
        console.log("Successfully File created");

    }
})
```

fs.readFile()

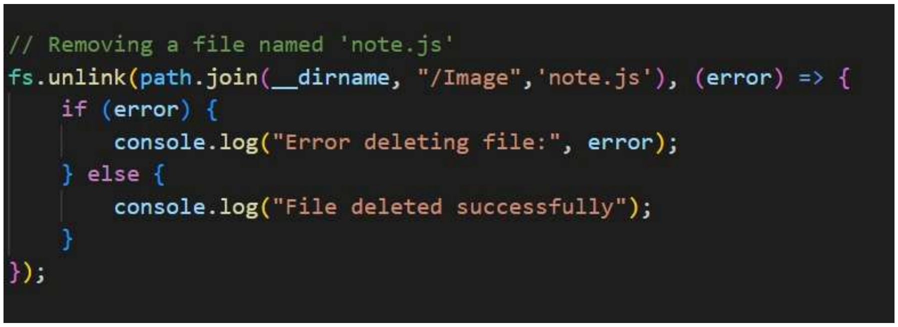
It is used to read the file.



```
fs.readFile(path.join(__dirname,'/Image','abcd.txt'),'utf-8',(error,data)=>{
    if (error) {
        console.log(error);
    }
    else{
        console.log(data);
    }
})
```

fs.unlink()

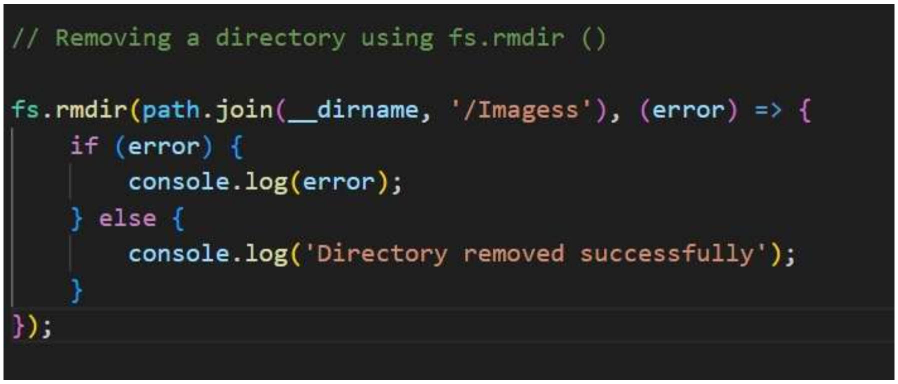
It is used to delete file



```
// Removing a file named 'note.js'
fs.unlink(path.join(__dirname, "/Image",'note.js'), (error) => {
    if (error) {
        console.log("Error deleting file:", error);
    } else {
        console.log("File deleted successfully");
    }
});
```

fs.rmdir()

It is used to delete a directory at the given path



```
// Removing a directory using fs.rmdir ()

fs.rmdir(path.join(__dirname, '/Imagess'), (error) => {
    if (error) {
        console.log(error);
    } else {
        console.log('Directory removed successfully');
    }
});
```