

1. What is NodeJS?

The NodeJS is an open source server environment which allows to run the JavaScript on the server side with Object Oriented Programming concepts. It has different important features.

- NodeJS runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- NodeJS can collect form data from the client and can generate dynamic contents
- NodeJS can add, delete, modify and read the data with different databases like MySQL, MongoDB etc.
- It can automatically create a web server without need of any extra web server like Apache, Internet Information Server.
- All NodeJS file get saved with .js extension
- NodeJS works on asynchronous or nonblockching or callback programming model.

2. What is meant by asynchronous programming?

Most of the web technologies like PHP, ASP, JSP etc. work on synchronous programming model in which when a client sends a request to the web server, the server sends that request to the file system, waits till file system reads and opens the file, send the response to the client and ready for the next request. The web server is busy till the request is not fulfilled by the file system.

But NodeJS works in asynchronous programming model in which when a client sends a request to the web server, the server sends the task to the web server and ready for the next request. When the file system open and read the file, the server returns the response to the client. The web server is available to many clients without waiting to complete the request.

Using asynchronous programming model, NodeJS eliminates the waiting, and simply continues with the next request.

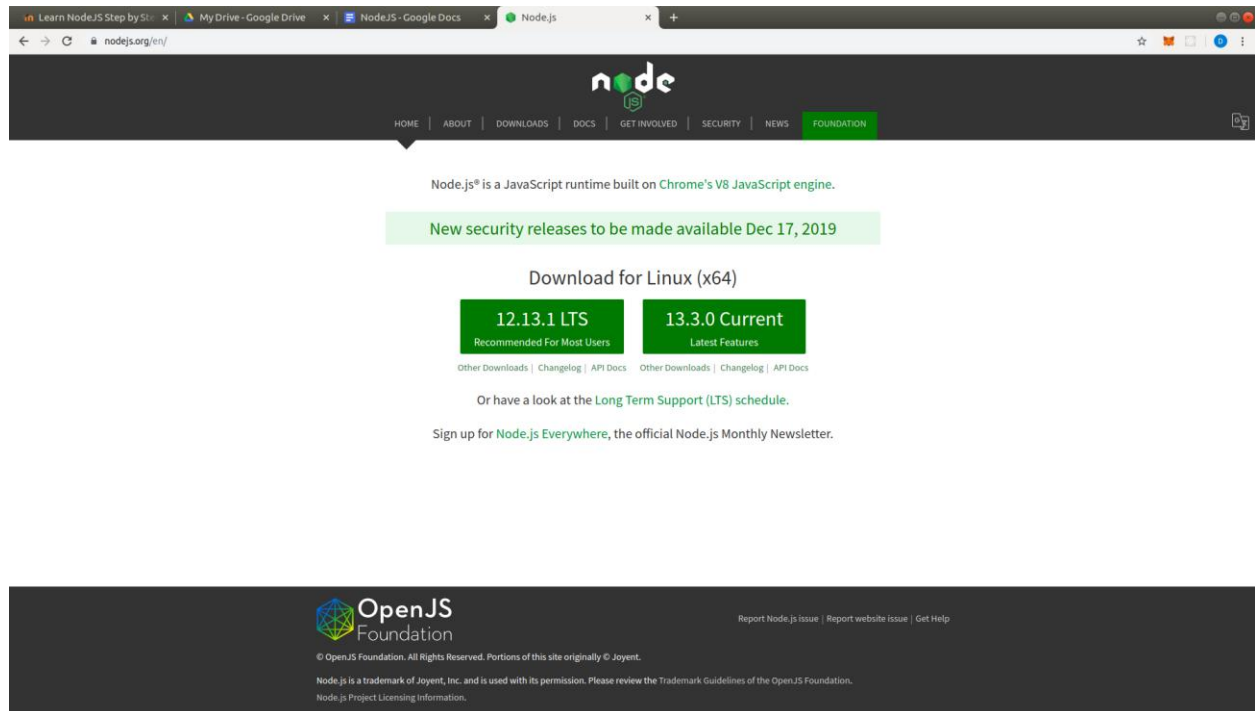
NodeJS runs single-threaded, non-blocking, asynchronous programming model, which is very memory efficient.

Better will be clear in the next section with examples. But you need to keep patience for it.

3. What are the prerequisites to use the NodeJS?

To start with NodeJS you need to first install the NodeJS software on your machine from given URL based on your operating system

<https://nodejs.org/en/>



4. How to check whether NodeJS is properly installed or not?

When you install NodeJS software, it installs three tools required for different tasks to be discussed later

- node
- npm
- npx

Check the versions of your tools use -v options, it verifies that installation is correct

Issue the following commands on your command prompt and see the version number of your installation.

node -v

npm -v

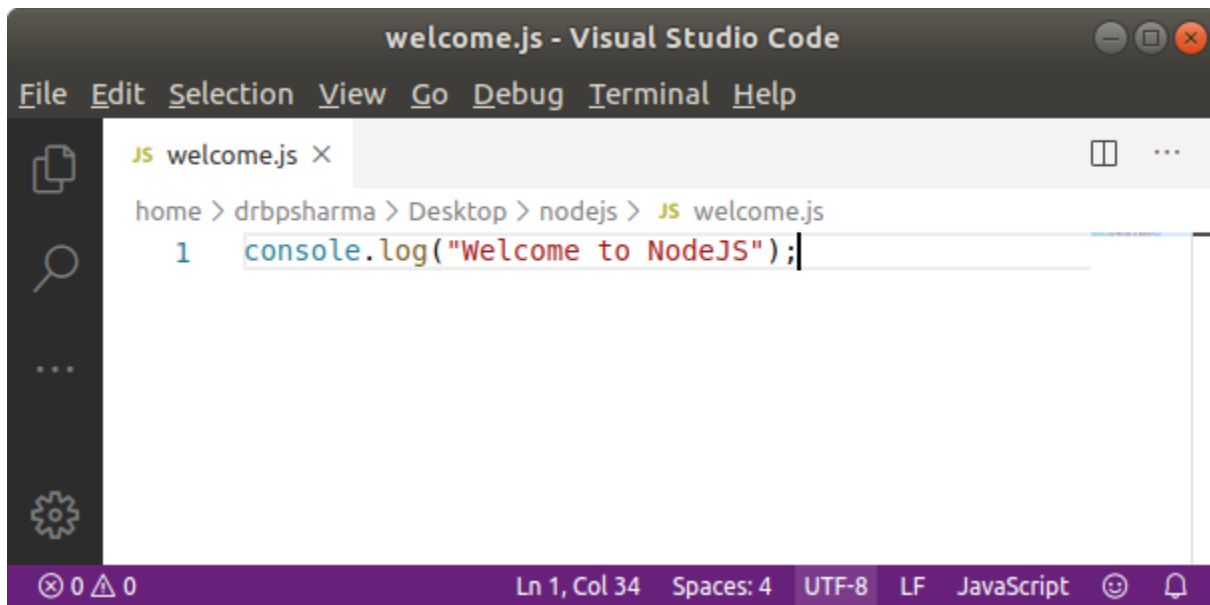
npx -v

5. How to send output to the console in NodeJS?

NodeJS provides a built-in object called console to interact with the console window. Use console.log() function to send some output to the console window.

Example

Create a JavaScript file as welcome.js with following contents and run that file using node command



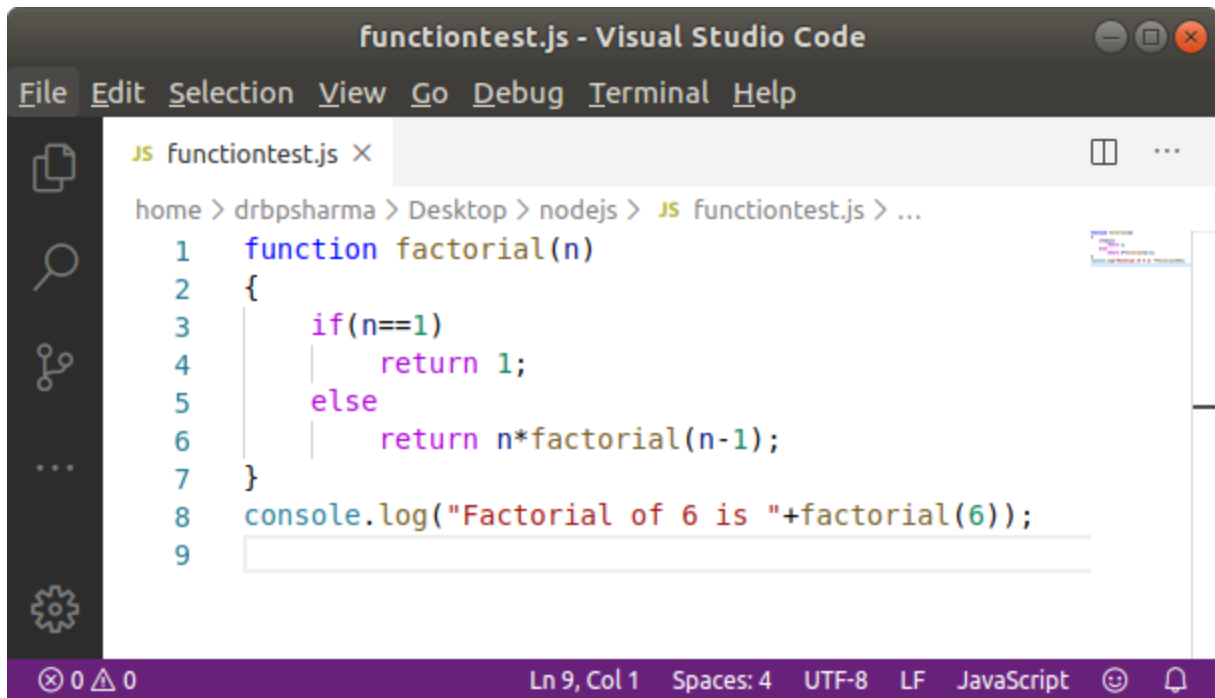
```
welcome.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
JS welcome.js x
home > drbpsharma > Desktop > nodejs > JS welcome.js
1 console.log("Welcome to NodeJS");
Ln 1, Col 34 Spaces: 4 UTF-8 LF JavaScript
```

6. How to create and use the functions in NodeJS?

Create the function like normal JavaScript code using function keyword and call them.

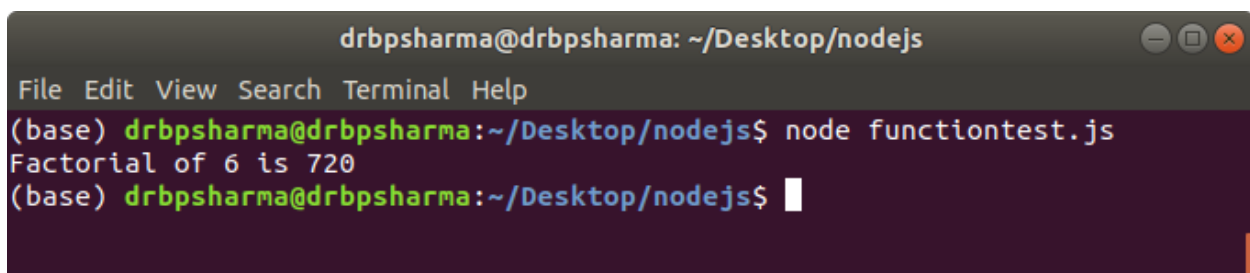
Example

Create a function factorial() which takes a number as argument and returns factorial of that number using recursion. Call that function to show factorial of 6.



```
functiontest.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help

JS functiontest.js X
home > drbpsharma > Desktop > nodejs > JS functiontest.js > ...
1 function factorial(n)
2 {
3     if(n==1)
4         return 1;
5     else
6         return n*factorial(n-1);
7 }
8 console.log("Factorial of 6 is "+factorial(6));
9
```



```
drbpsharma@drbpsharma: ~/Desktop/nodejs
File Edit View Search Terminal Help
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node functiontest.js
Factorial of 6 is 720
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$
```

7. How to create and use the classes and instances in NodeJS?

NodeJS allows to create the classes and instances.

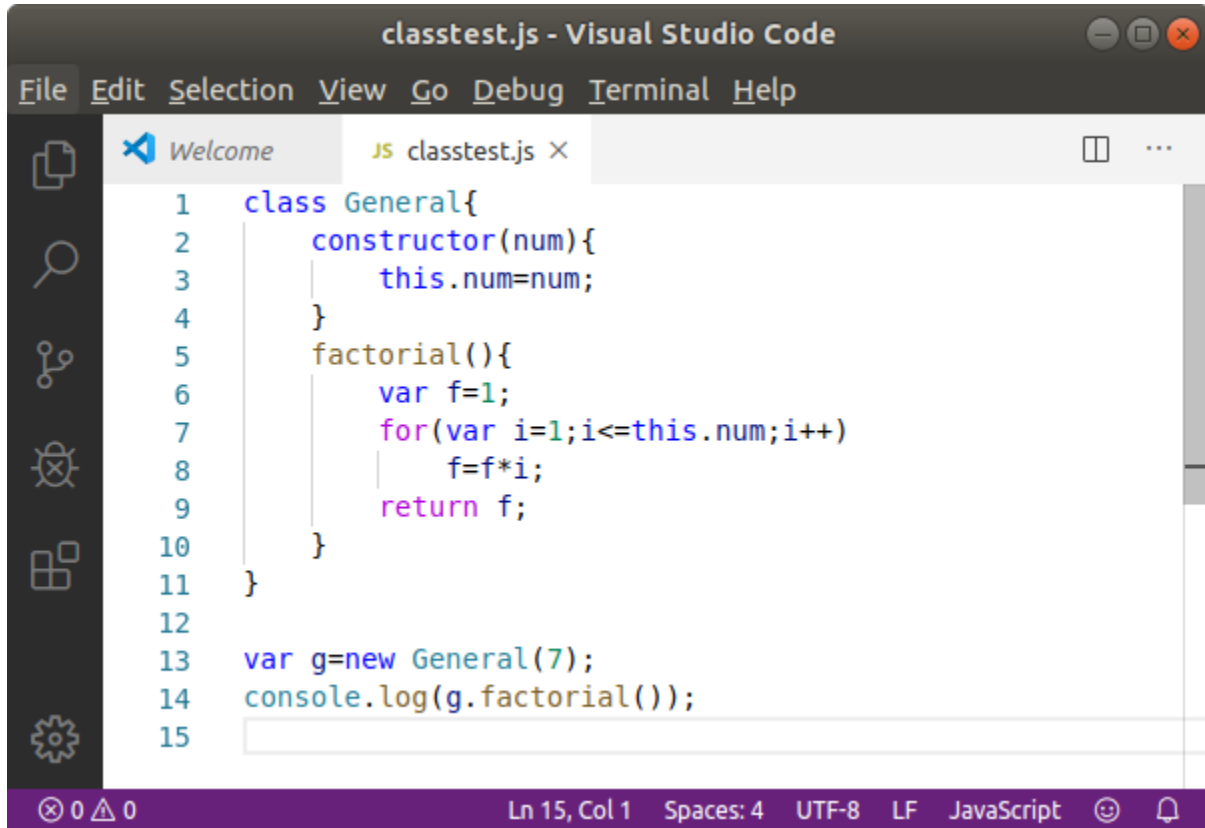
- Use class keyword to create a class
- Use constructor keyword to create a constructor
- Use this keyword to refer the current instance
- Use new keyword to create an instance

Example

Create a class General having a data member as num. Create a method factorial() to return factorial of num.

Try following steps

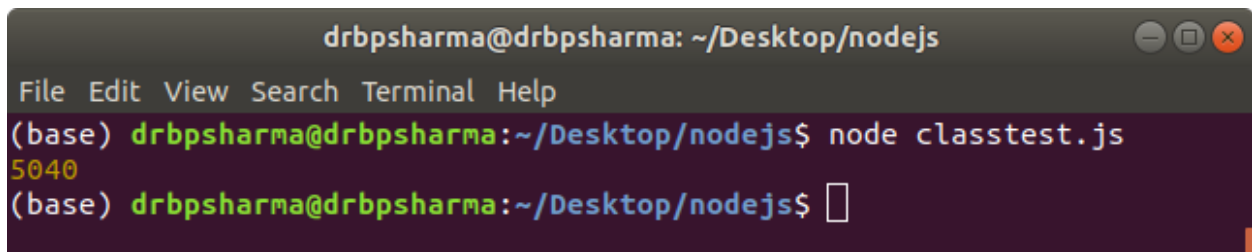
Write the following code using Visual Studio Code IDE and save it as classtest.js and run it on command prompt



The screenshot shows the Visual Studio Code editor with a file named 'classtest.js' open. The code is written in JavaScript and defines a 'General' class with a 'constructor' and a 'factorial' method. It then creates an instance 'g' of the 'General' class and logs the result of 'g.factorial()' to the console.

```
1  class General{
2      constructor(num){
3          this.num=num;
4      }
5      factorial(){
6          var f=1;
7          for(var i=1;i<=this.num;i++)
8              f=f*i;
9          return f;
10     }
11 }
12
13 var g=new General(7);
14 console.log(g.factorial());
15
```

The status bar at the bottom indicates the file is at line 15, column 1, with 4 spaces, using UTF-8 encoding and LF line endings, and is a JavaScript file.



The screenshot shows a terminal window with the prompt 'drbpsharma@drbpsharma: ~/Desktop/nodejs'. The user has entered the command 'node classtest.js' and the output '5040' is displayed.

```
drbpsharma@drbpsharma: ~/Desktop/nodejs
File Edit View Search Terminal Help
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node classtest.js
5040
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$
```

8. How we can create a library or module of functions and call that functions in any NodeJS Program?

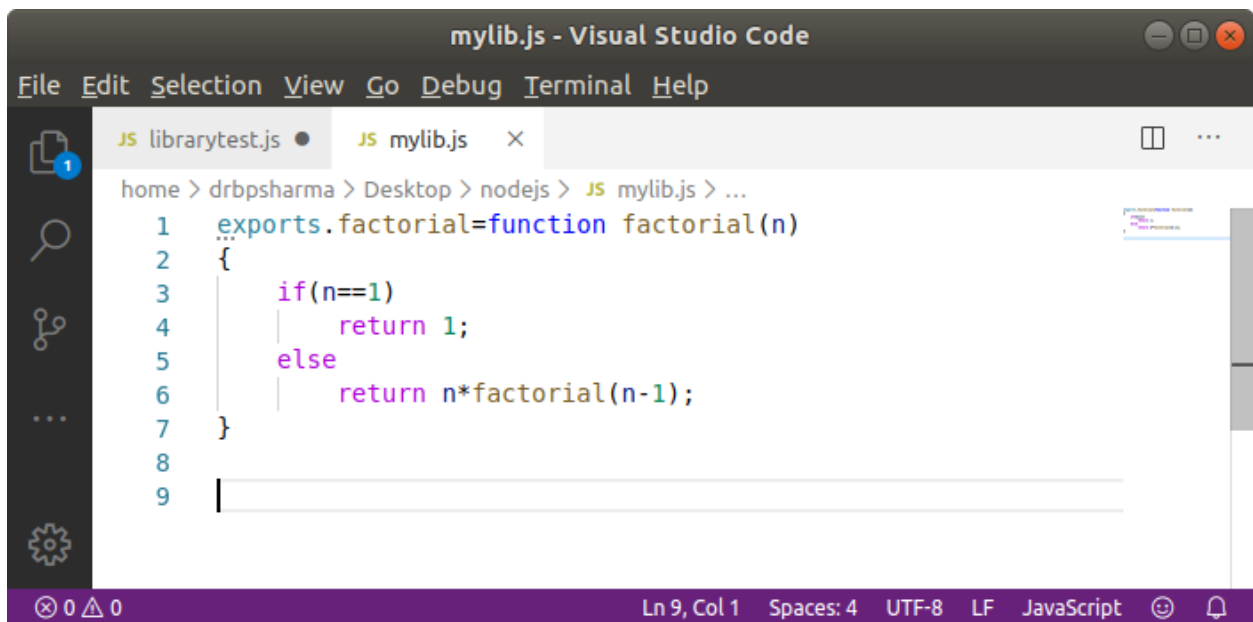
A library or module is a JavaScript file having a set of functions which can be reused in other programs as an when required.

To create a library or module, create a normal .js program having set of functions to be reused as library. Use exports command to make the function accessible to other programs as library.

Use require() function to import that library in any other program. Create a reference of the library using var keyword and call that function.

Example

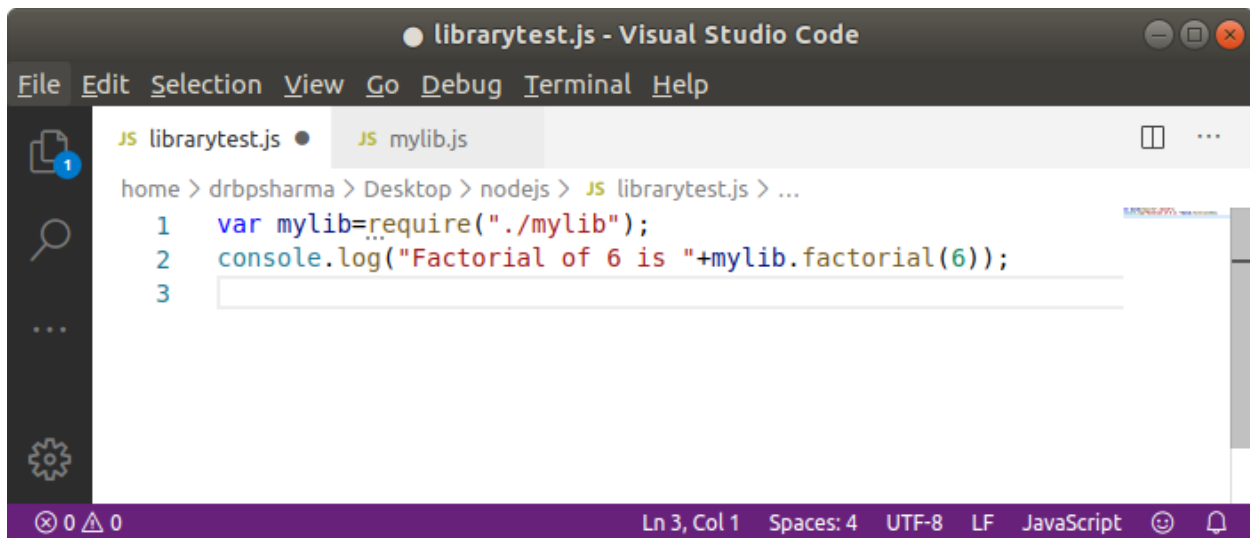
Create a library file as mylib.js having the recursive function factorial() which takes a number and returns factorial of that number. Use exports command to export that function.



The screenshot shows the Visual Studio Code editor with a file named `mylib.js` open. The editor has a dark theme and a sidebar on the left with icons for Explorer, Search, Source Control, and Settings. The top menu bar includes File, Edit, Selection, View, Go, Debug, Terminal, and Help. The file explorer shows two files: `librarytest.js` and `mylib.js`. The `mylib.js` file is selected and its content is displayed in the editor. The code defines a recursive function `factorial` and exports it using `exports.factorial`. The status bar at the bottom indicates the current position is Line 9, Column 1, with 4 spaces, UTF-8 encoding, LF line endings, and the JavaScript language.

```
mylib.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
JS librarytest.js JS mylib.js
home > drbpsharma > Desktop > nodejs > JS mylib.js > ...
1 exports.factorial=function factorial(n)
2 {
3     if(n==1)
4         return 1;
5     else
6         return n*factorial(n-1);
7 }
8
9
```

Now create another JavaScript program librarytest.js to import that module using require() function and call the function factorial() to show factorial of 6.

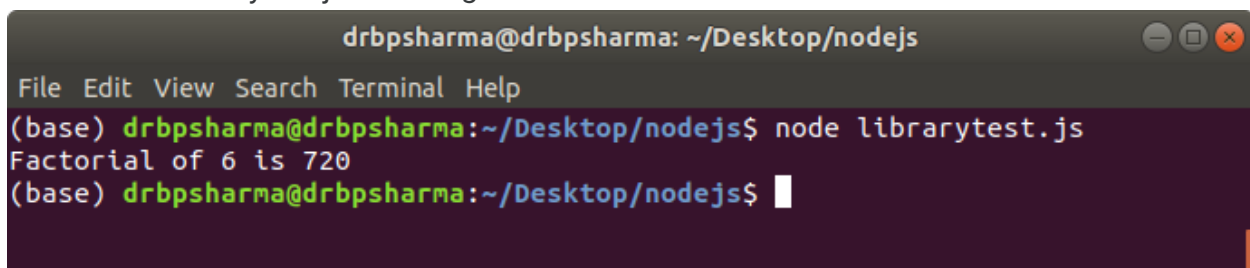


The screenshot shows the Visual Studio Code interface. The title bar reads "librarytest.js - Visual Studio Code". The menu bar includes File, Edit, Selection, View, Go, Debug, Terminal, and Help. The Explorer sidebar on the left shows two files: librarytest.js and mylib.js. The main editor area displays the content of librarytest.js, which is:

```
home > drbpsharma > Desktop > nodejs > JS librarytest.js > ...  
1 var mylib=require("./mylib");  
2 console.log("Factorial of 6 is "+mylib.factorial(6));  
3
```

 The status bar at the bottom indicates "Ln 3, Col 1", "Spaces: 4", "UTF-8", "LF", and "JavaScript".

Now run the librarytest.js file using node command



The screenshot shows a terminal window with the title "drbpsharma@drbpsharma: ~/Desktop/nodejs". The terminal output is:

```
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node librarytest.js  
Factorial of 6 is 720  
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$
```

9. What are different types of modules in NodeJS?

Modules can be of three types

1. User Defined Modules
2. Predefined Modules
3. Third Party Modules

User defined modules are created by the developers as per their requirements as we did in the last example. Here we need to specify the path of the module file.

Example

```
var mylib=require('./mylib');
```

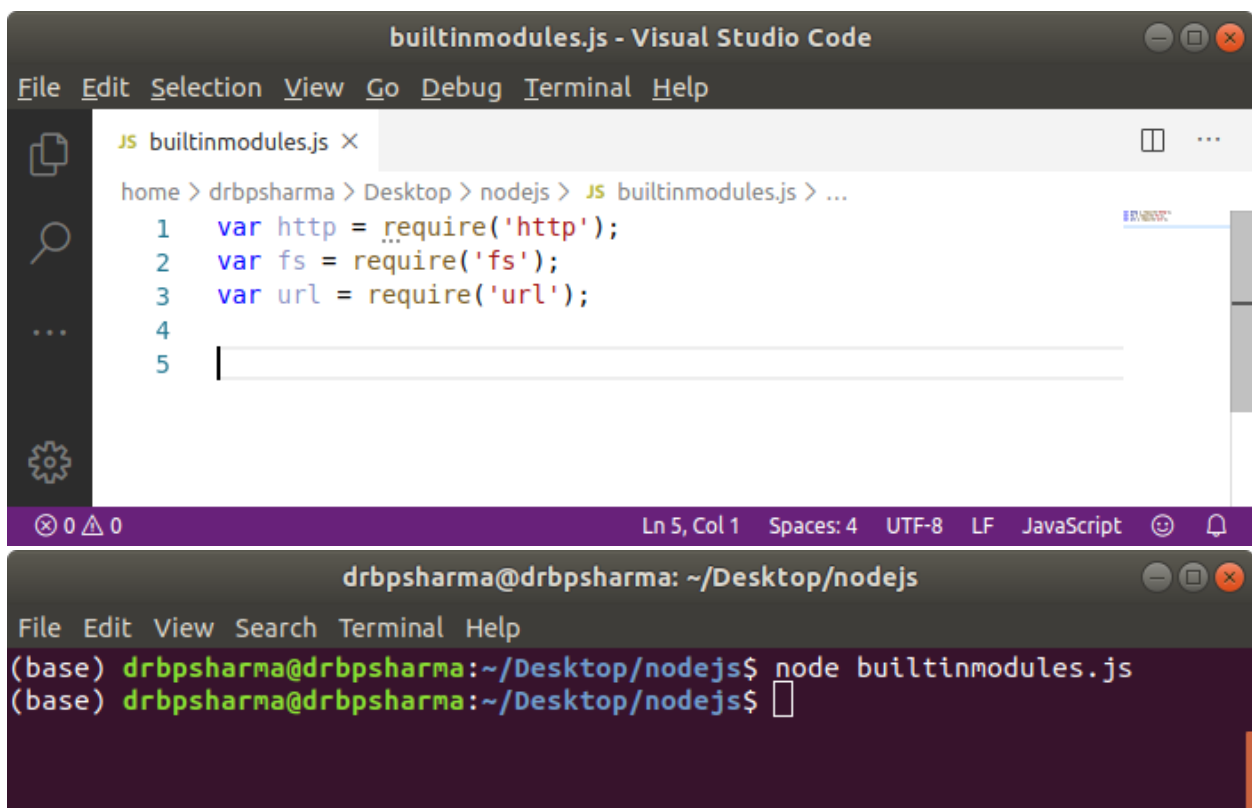
Predefined modules are supplied with NodeJS software and can be imported directly whenever we need them without specifying their path. Some of the predefined modules in NodeJS are

- http
- fs
- Url

If you import these modules in your program they will be imported without any error. Use of these modules will be discussed later.

Example

Create a file builtinmodules.js and import all three modules in it and run it. No error occurs.



The image shows two windows. The top window is Visual Studio Code, titled 'builtinmodules.js - Visual Studio Code'. It displays a JavaScript file with the following code:

```
1 var http = require('http');
2 var fs = require('fs');
3 var url = require('url');
4
5 |
```

The bottom window is a terminal, titled 'drbpsharma@drbpsharma: ~/Desktop/nodejs'. It shows the command 'node builtinmodules.js' being executed, which runs successfully without any output or errors.

Third party modules are installed on demand using special software provided with NodeJS called as npm (Node Package Manager).

For example, if you want to send an email using NodeJS we need nodemailer module, to use the mysql database with NodeJS we need mysql module, to use the MongoDB database with NodeJS we need mongodb library.

Use the following commands to import these modules into your machine.

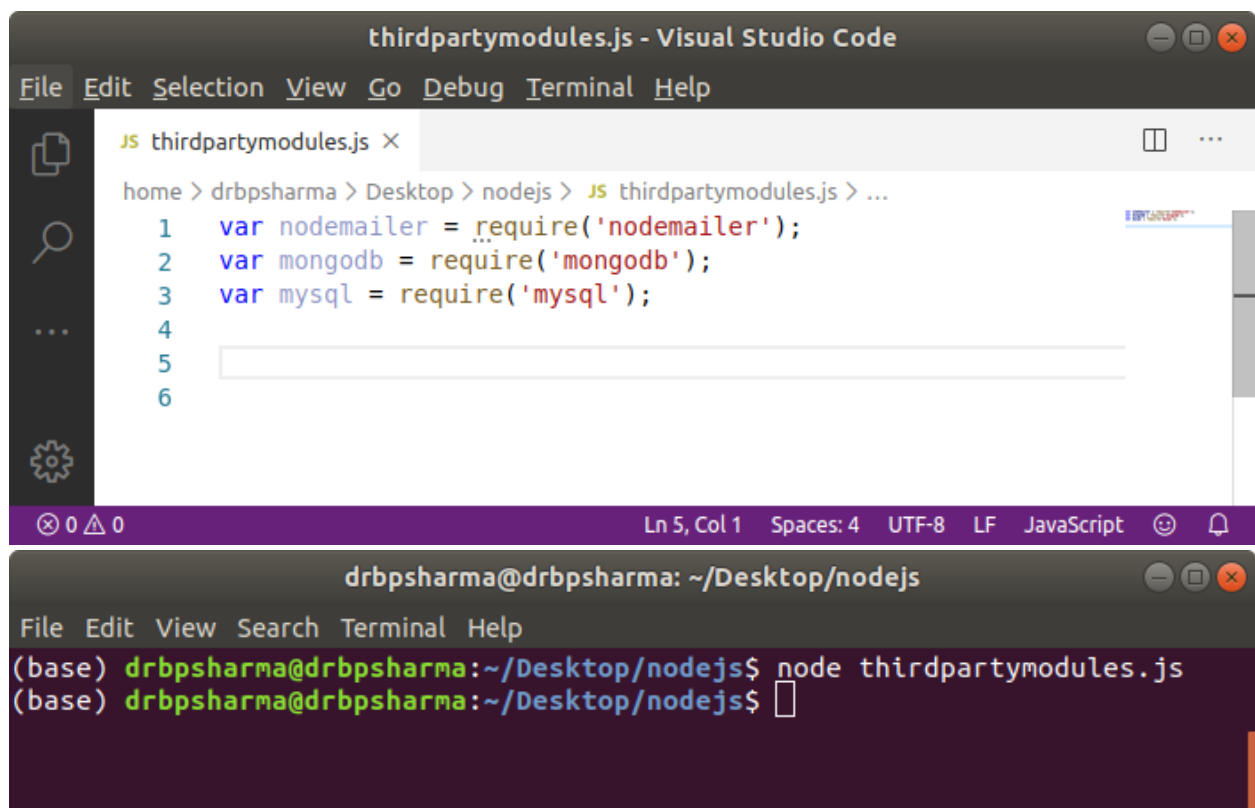

```
npm install -g nodemailer
```

```
npm install -g mysql
```

```
npm install -g mongodb
```

Now we can also import these modules.

Example



The image shows a Visual Studio Code editor window titled 'thirdpartymodules.js - Visual Studio Code'. The editor displays a JavaScript file named 'thirdpartymodules.js' with the following code:

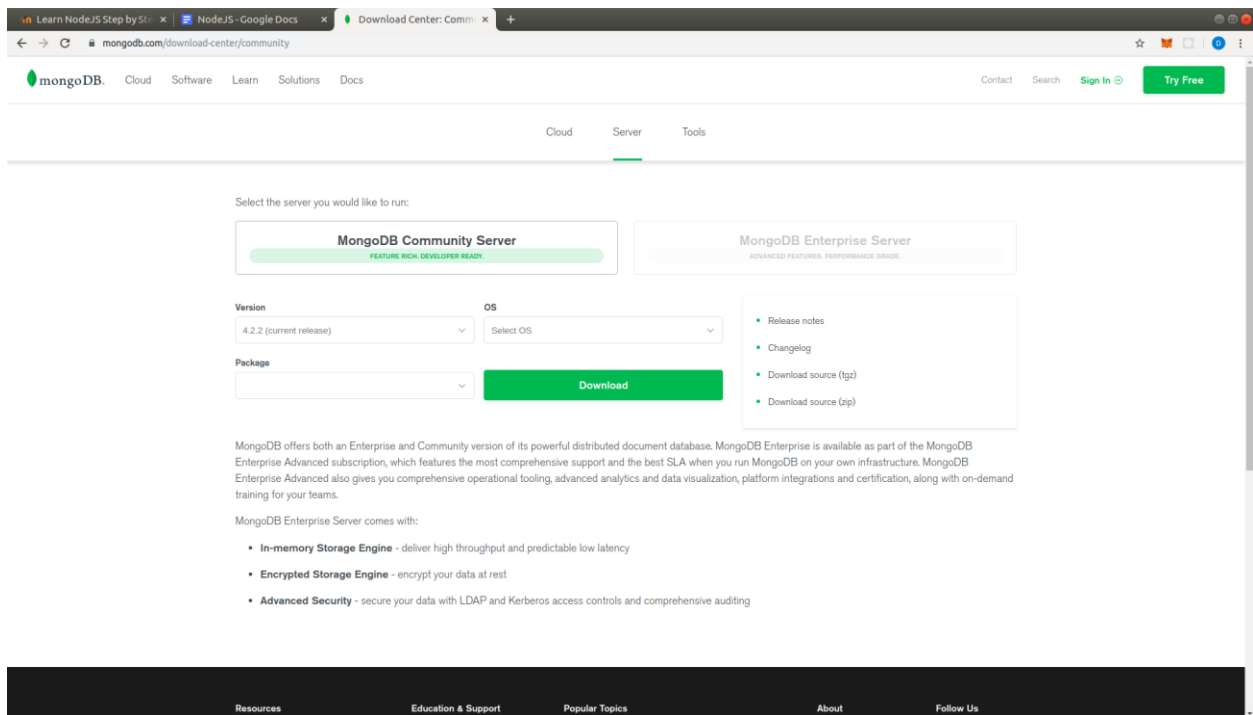
```
1 var nodemailer = require('nodemailer');
2 var mongodb = require('mongodb');
3 var mysql = require('mysql');
4
5
6
```

Below the editor is a terminal window titled 'drbpsharma@drbpsharma: ~/Desktop/nodejs'. The terminal shows the command 'node thirdpartymodules.js' being executed, and the prompt returns to the shell.

10. How to install MongoDB Server on local machine?

Use community edition of MongoDB server to download and install it on local machine from

<https://www.mongodb.com/download-center/community>



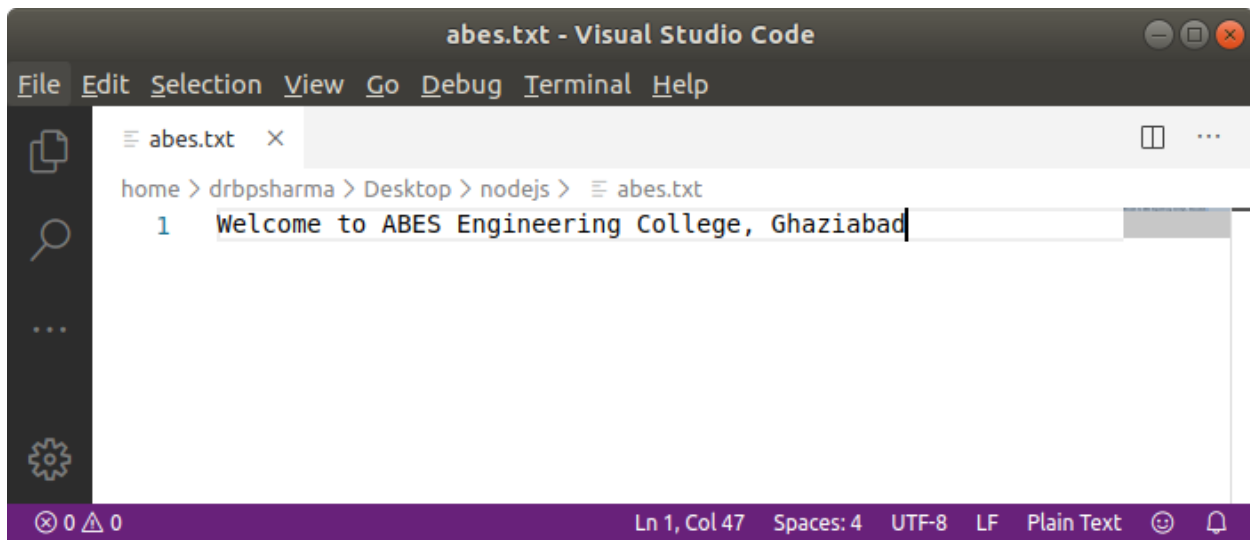
11. What is meant by blocking code?

A code which waits till some given task get complete and then moves to the next statement is called as blocking code.

It such case program get stuck at some point till the task in processing, It a task is taking a long time then program will not move to the next statement.

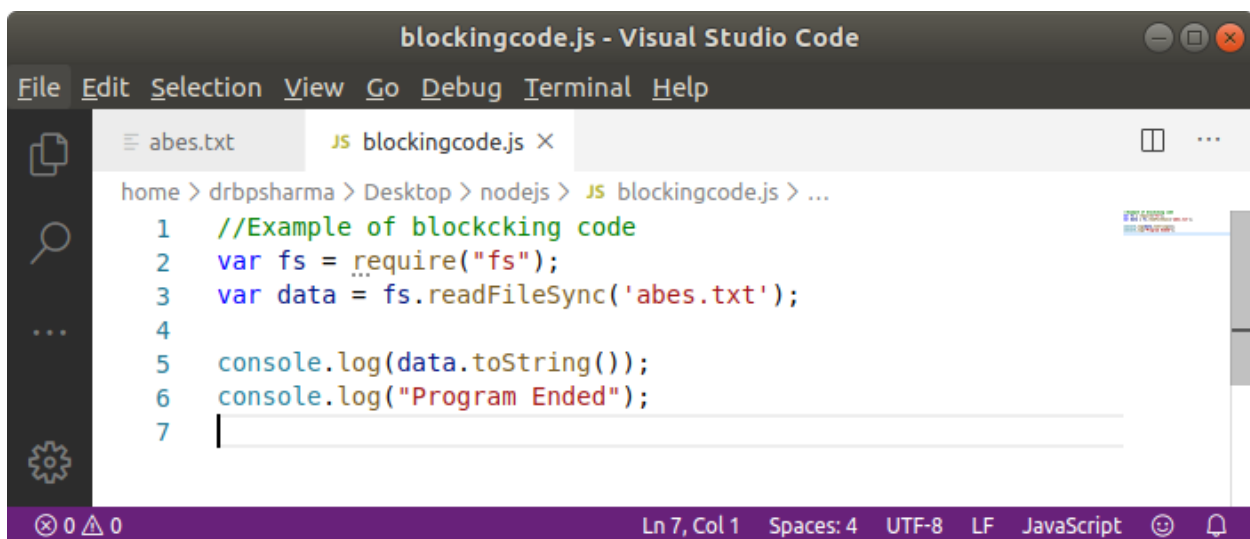
Example

Create a text file as abes.txt having some text like



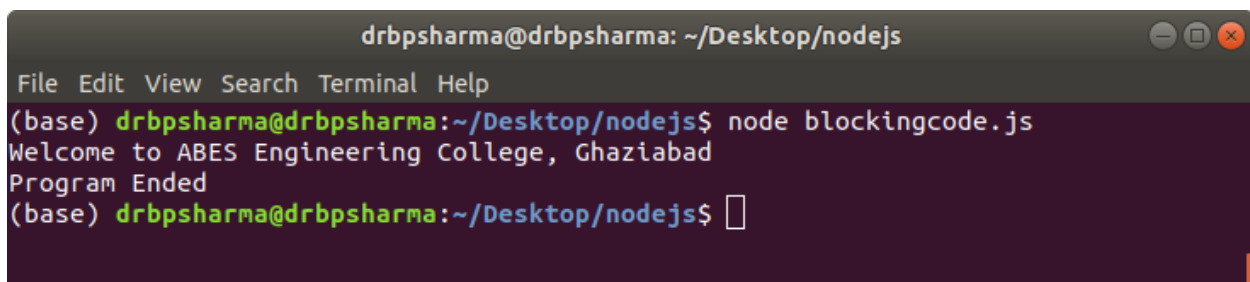
```
abes.txt - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
abes.txt x
home > drbpsharma > Desktop > nodejs > abes.txt
1 Welcome to ABES Engineering College, Ghaziabad
Ln 1, Col 47 Spaces: 4 UTF-8 LF Plain Text
```

Now write the following code to read the contents of the file abes.txt and show that contents on the screen



```
blockingcode.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
abes.txt JS blockingcode.js x
home > drbpsharma > Desktop > nodejs > JS blockingcode.js > ...
1 //Example of blockcking code
2 var fs = require("fs");
3 var data = fs.readFileSync('abes.txt');
4
5 console.log(data.toString());
6 console.log("Program Ended");
7
Ln 7, Col 1 Spaces: 4 UTF-8 LF JavaScript
```

Run the code



```
drbpsharma@drbpsharma: ~/Desktop/nodejs
File Edit View Search Terminal Help
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node blockingcode.js
Welcome to ABES Engineering College, Ghaziabad
Program Ended
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$
```

This program code tries to read the data from the text file and blocks the execution till the data is read, display the data then moves to the next statement.

It is called as synchronous or blocking code execution process which is generally used by most of the server side technologies like JSP, PHP, ASP etc.

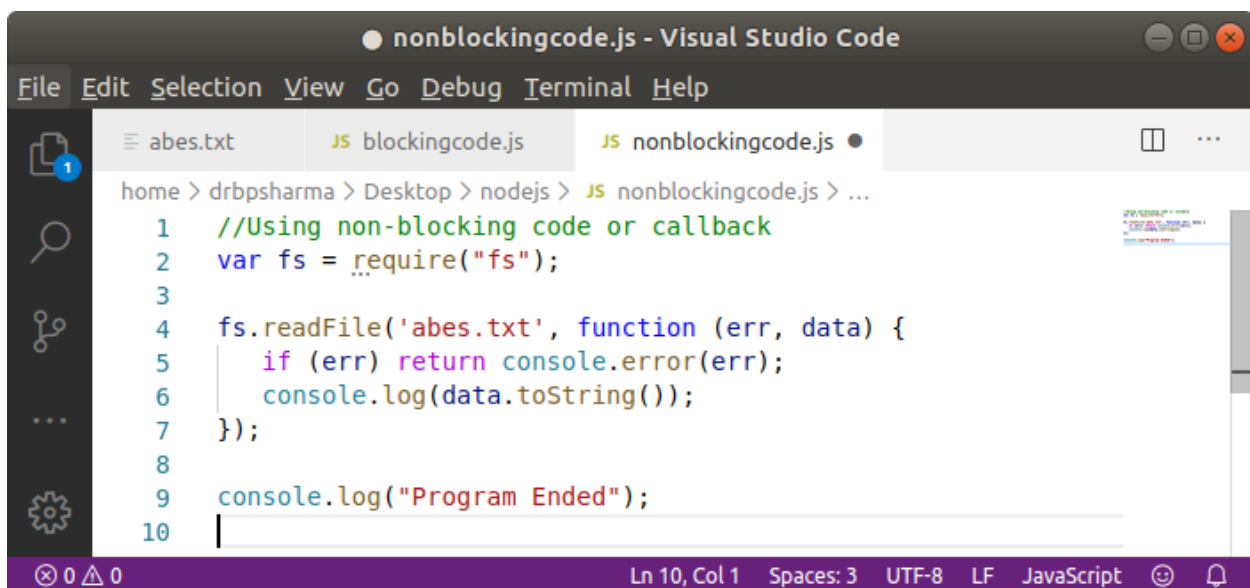
12. What is non-blocking code or callback?

A code which continue to execute to next statement even if the task is in process. Once the task get complete it returns but the code is not halted at that statement. This method is also called as callback.

Example

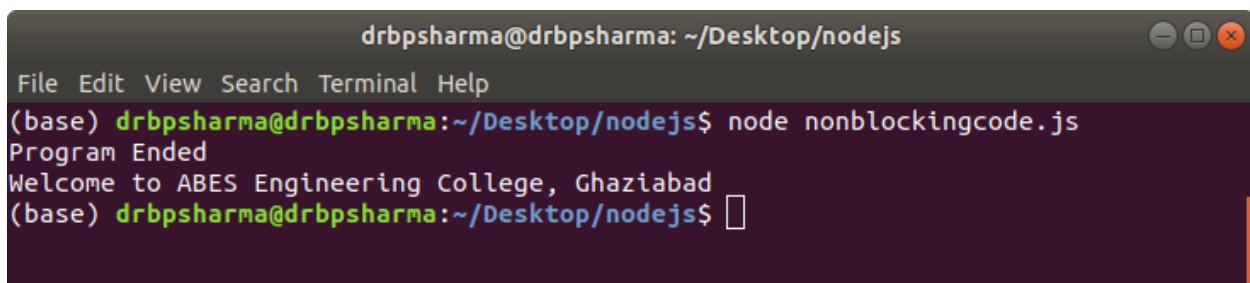
Use the text file abes.txt created in previous example.

Write the following code to test the non-blocking code



```
nonblockingcode.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
abes.txt JS blockingcode.js JS nonblockingcode.js
home > drbpsharma > Desktop > nodejs > JS nonblockingcode.js > ...
1 //Using non-blocking code or callback
2 var fs = require("fs");
3
4 fs.readFile('abes.txt', function (err, data) {
5     if (err) return console.error(err);
6     console.log(data.toString());
7 });
8
9 console.log("Program Ended");
10
```

Now execute it and see the output



```
drbpsharma@drbpsharma: ~/Desktop/nodejs
File Edit View Search Terminal Help
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node nonblockingcode.js
Program Ended
Welcome to ABES Engineering College, Ghaziabad
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$
```

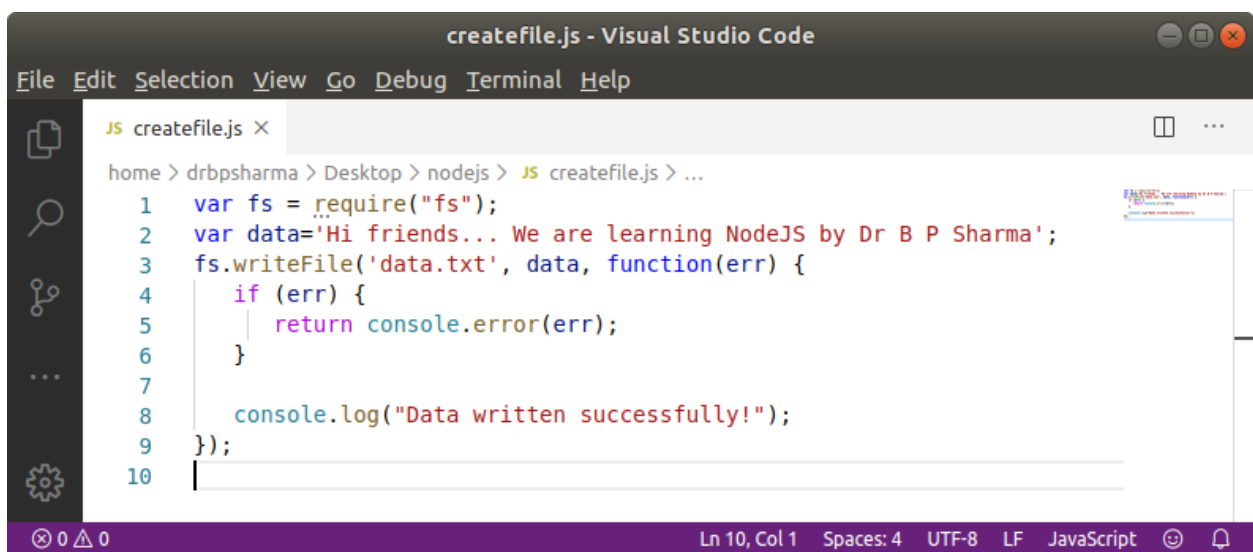
Here the file is in read process and result is send back when available but program moves to next statement without waiting to complete the previous task.

Such code is known as non-blocking code or asynchronous code or callback which is power of NodeJS.

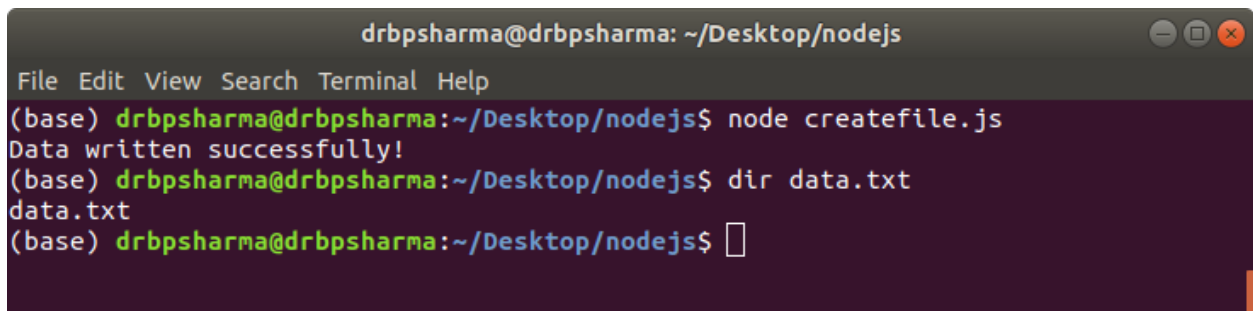
Whenever we say to use the callback in some syntax, you need to use this syntax.

13. How to create a file using NodeJS?

To write contents to some file use writeFile() function of fs module.

A screenshot of the Visual Studio Code editor window titled 'createfile.js - Visual Studio Code'. The editor shows a JavaScript file named 'createfile.js' with the following code:

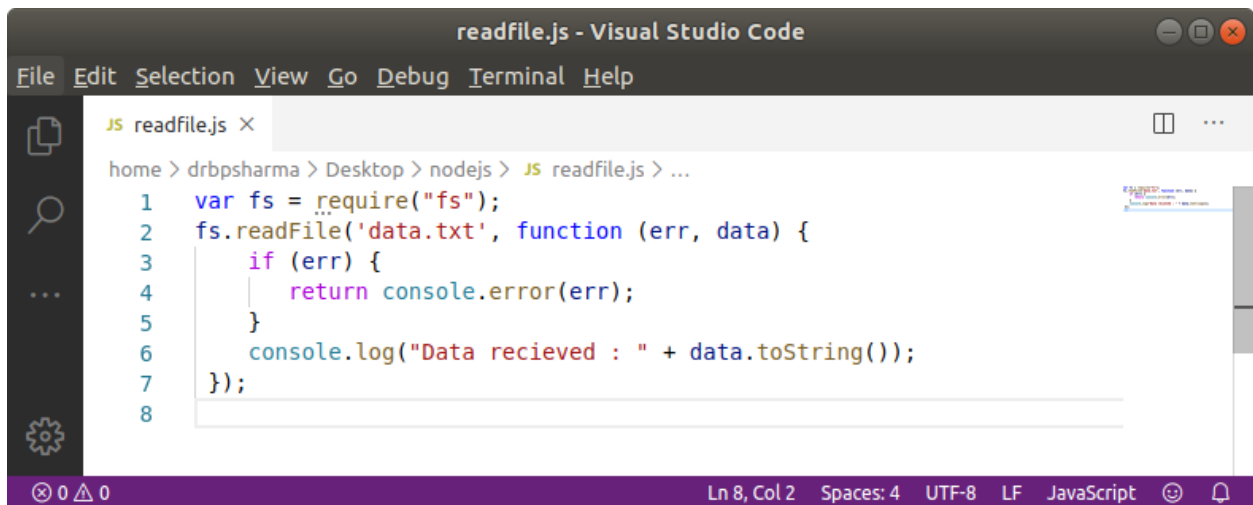
```
1 var fs = require("fs");
2 var data='Hi friends... We are learning NodeJS by Dr B P Sharma';
3 fs.writeFile('data.txt', data, function(err) {
4     if (err) {
5         return console.error(err);
6     }
7
8     console.log("Data written successfully!");
9 });
10
```

The status bar at the bottom indicates 'Ln 10, Col 1', 'Spaces: 4', 'UTF-8', 'LF', and 'JavaScript'.A screenshot of a terminal window titled 'drbpsharma@drbpsharma: ~/Desktop/nodejs'. The terminal shows the following commands and output:

```
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node createfile.js
Data written successfully!
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ dir data.txt
data.txt
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$
```

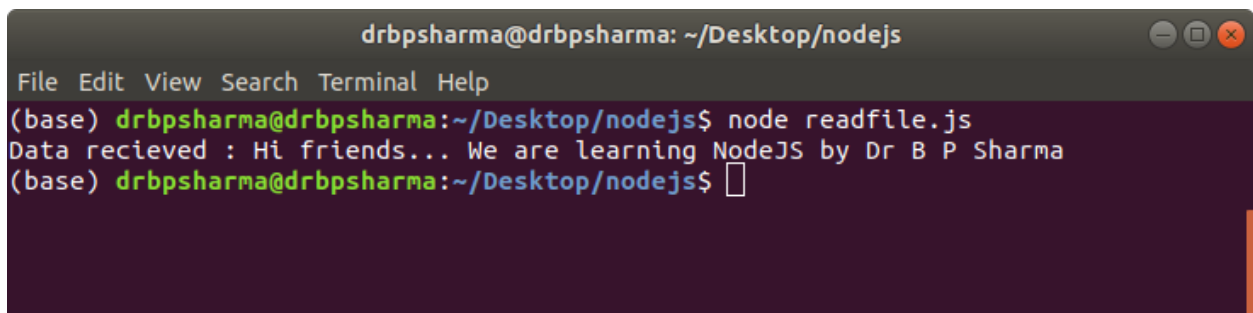
14. How to read the contents of a file using NodeJS?

Use readFile() function of fs module to read the contents of the file.



```
readfile.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help

JS readFile.js x
home > drbpsharma > Desktop > nodejs > JS readFile.js > ...
1 var fs = require("fs");
2 fs.readFile('data.txt', function (err, data) {
3     if (err) {
4         return console.error(err);
5     }
6     console.log("Data recieved : " + data.toString());
7 });
8
```



```
drbpsharma@drbpsharma: ~/Desktop/nodejs
File Edit View Search Terminal Help
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node readFile.js
Data recieved : Hi friends... We are learning NodeJS by Dr B P Sharma
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$
```

15. How to know the properties of a file?

Use stat() function of fs module to know the statistics (properties) of a file like file size, date of creation etc.



```
showfileinfo.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help

JS showfileinfo.js x
home > drbpsharma > Desktop > nodejs > JS showfileinfo.js > ...
1 var fs = require("fs");
2 fs.stat('data.txt', function (err, stats) {
3     if (err) {
4         return console.error(err);
5     }
6     console.log(stats);
7 });
8
```

```
drbpsharma@drbpsharma: ~/Desktop/nodejs
File Edit View Search Terminal Help
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node showfileinfo
Stats {
  dev: 66306,
  mode: 33188,
  nlink: 1,
  uid: 1000,
  gid: 1000,
  rdev: 0,
  blksize: 4096,
  ino: 23080763,
  size: 53,
  blocks: 8,
  atimeMs: 1575654157725.7837,
  mtimeMs: 1575653908802.5486,
  ctimeMs: 1575653908802.5486,
  birthtimeMs: 1575653810817.01,
  atime: 2019-12-06T17:42:37.726Z,
  mtime: 2019-12-06T17:38:28.803Z,
  ctime: 2019-12-06T17:38:28.803Z,
  birthtime: 2019-12-06T17:36:50.817Z
}
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$
```

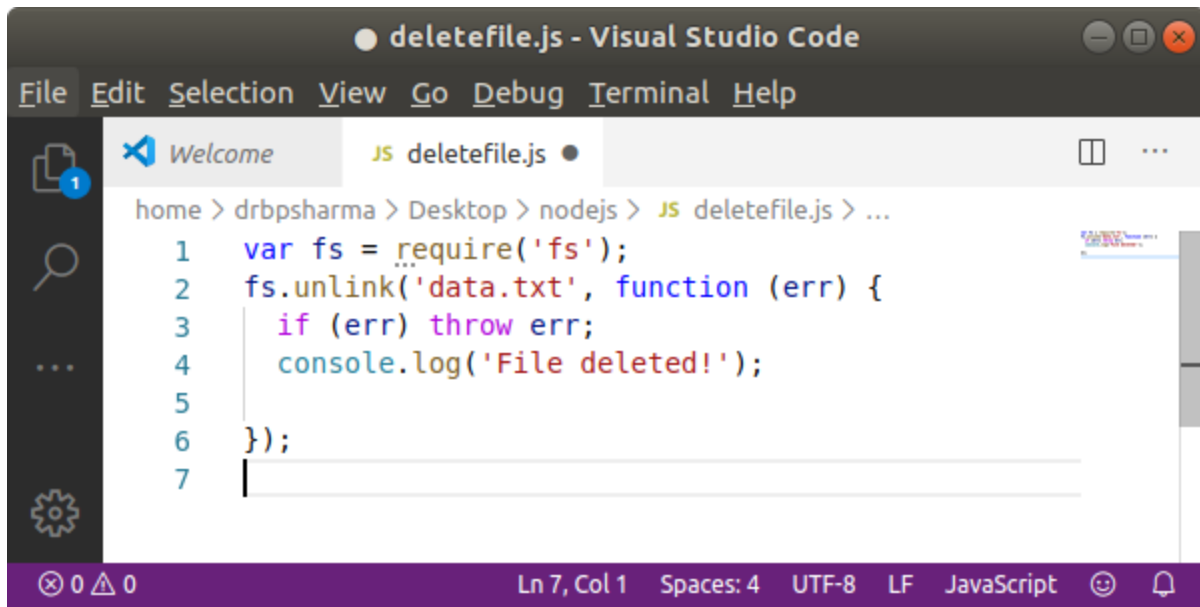
16. How to delete a file using NodeJS?

To delete a file use the `unlink()` method of `fs` module with a callback.

Syntax

```
fs.unlink('filename',callback);
```

Try the following code



```
deletefile.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help

Welcome JS deletefile.js

home > drbpsharma > Desktop > nodejs > JS deletefile.js > ...
1 var fs = require('fs');
2 fs.unlink('data.txt', function (err) {
3   if (err) throw err;
4   console.log('File deleted!');
5
6 });
7
```

Ln 7, Col 1 Spaces: 4 UTF-8 LF JavaScript

18. How to use MySQL with NodeJS?

To store data in database using NodeJS we can use almost any database but MySQL and MongoDB are most preferred one.

To use the MySQL database with NodeJS we need to import the mysql module into your machine using Node Package Manager (npm)

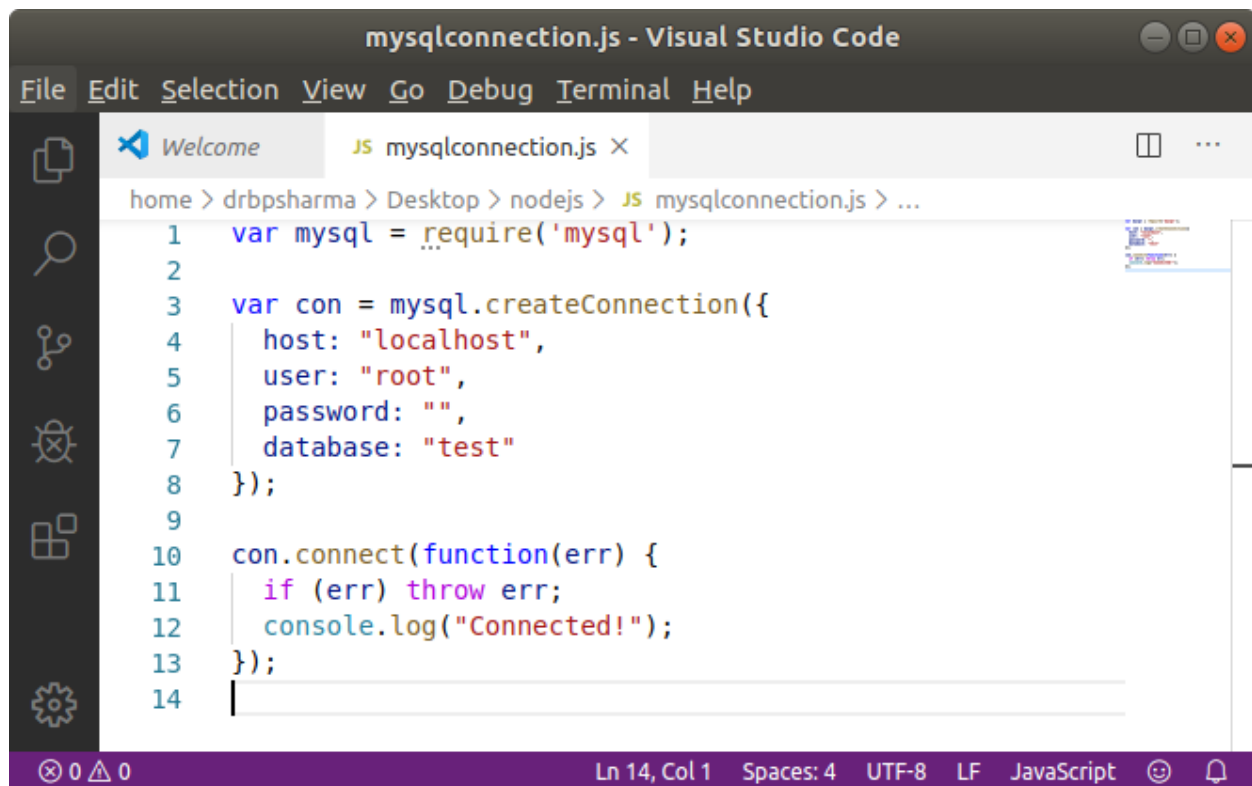
```
npm install -g mysql
```

Now you can use `require()` function to import the library and its functions.

Use `createConnection()` method of mysql module to define the parameters like host, user, password and database.

Use `connect()` method with a callback to establish the connection with database server and use it for CRUD operations.

Try the following code



```
mysqlconnection.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help

Welcome JS mysqlconnection.js X

home > drbpsharma > Desktop > nodejs > JS mysqlconnection.js > ...
1 var mysql = require('mysql');
2
3 var con = mysql.createConnection({
4   host: "localhost",
5   user: "root",
6   password: "",
7   database: "test"
8 });
9
10 con.connect(function(err) {
11   if (err) throw err;
12   console.log("Connected!");
13 });
14 |
```

Ln 14, Col 1 Spaces: 4 UTF-8 LF JavaScript

19. How to insert a record into MySQL table using NodeJS?

Use query() method from connection type reference of mysql module.

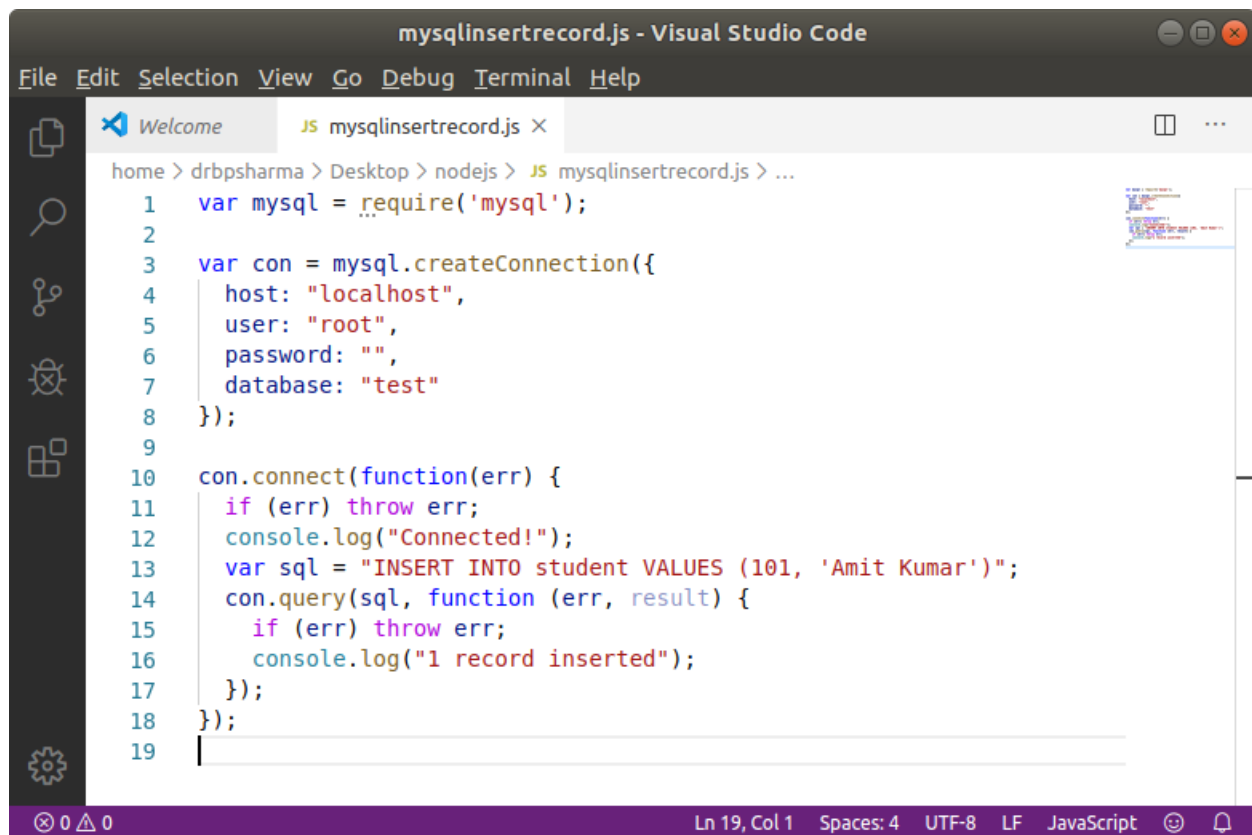
Sample Case:

Database: test

Table: student

Columns: rollno int, name varchar(50)

Try the following code



```
mysqlinsertrecord.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help

Welcome JS mysqlinsertrecord.js x

home > drbpsharma > Desktop > nodejs > JS mysqlinsertrecord.js > ...
1 var mysql = require('mysql');
2
3 var con = mysql.createConnection({
4   host: "localhost",
5   user: "root",
6   password: "",
7   database: "test"
8 });
9
10 con.connect(function(err) {
11   if (err) throw err;
12   console.log("Connected!");
13   var sql = "INSERT INTO student VALUES (101, 'Amit Kumar')";
14   con.query(sql, function (err, result) {
15     if (err) throw err;
16     console.log("1 record inserted");
17   });
18 });
19
```

Ln 19, Col 1 Spaces: 4 UTF-8 LF JavaScript

20. How to read records from MySQL table?

Use query() method of mysql connection reference with a callback to receive data against a SELECT query.

Try the following code

```
mysqlreadrecords.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help

Welcome JS mysqlreadrecords.js X

home > drbpsharma > Desktop > nodejs > JS mysqlreadrecords.js > ...
1 var mysql = require('mysql');
2
3 var con = mysql.createConnection({
4   host: "localhost",
5   user: "root",
6   password: "",
7   database: "test"
8 });
9
10 con.connect(function(err) {
11   if (err) throw err;
12   console.log("Connected!");
13   var sql = "SELECT * FROM student";
14   con.query(sql, function (err, result, fields) {
15     if (err) throw err;
16     console.log(result);
17   });
18 });
19
```

Ln 19, Col 1 Spaces: 4 UTF-8 LF JavaScript

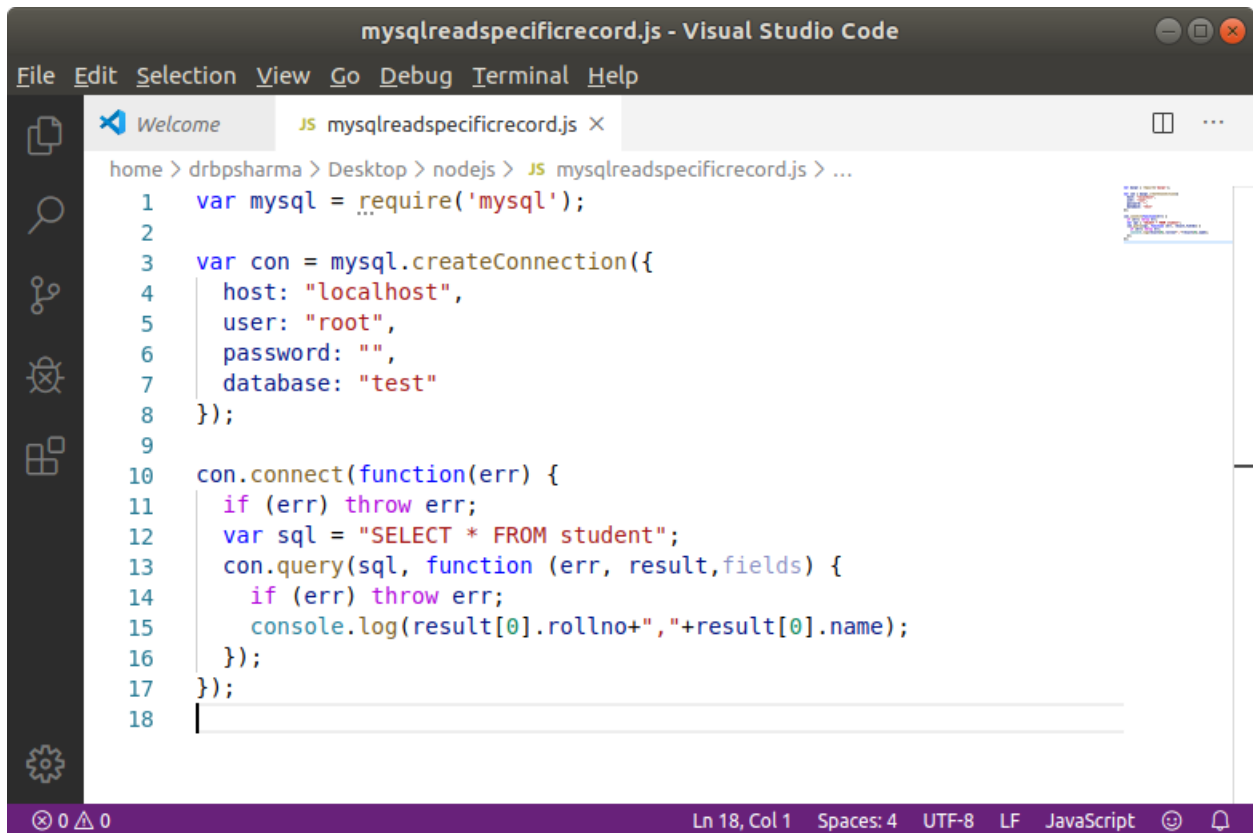
Output Screen

```
drbpsharma@drbpsharma: ~/Desktop/nodejs
File Edit View Search Terminal Help
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node mysqlreadrecords.js
Connected!
[
  RowDataPacket { rollno: 101, name: 'Amit Kumar' },
  RowDataPacket { rollno: 102, name: 'Nitin Kumar' },
  RowDataPacket { rollno: 103, name: 'Kamal Gupta' }
]
```

21. How to read specific record from the result in MySQL?

The result returned by the callback is an array. You can read specific record using an index number. You can also specify the column name to be accessed.

Try the following example



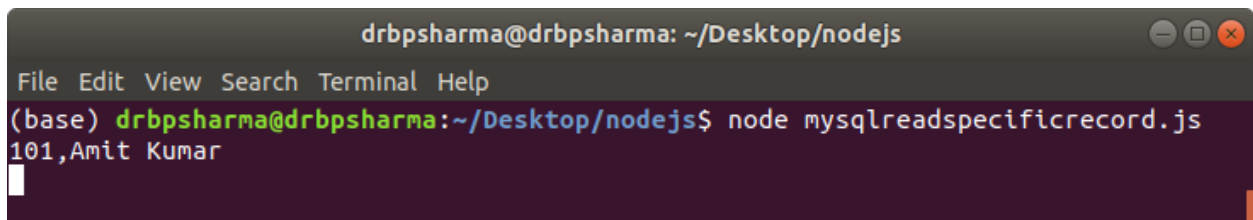
```
mysqlreadspecificrecord.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help

Welcome JS mysqlreadspecificrecord.js X

home > drbpsharma > Desktop > nodejs > JS mysqlreadspecificrecord.js > ...
1 var mysql = require('mysql');
2
3 var con = mysql.createConnection({
4   host: "localhost",
5   user: "root",
6   password: "",
7   database: "test"
8 });
9
10 con.connect(function(err) {
11   if (err) throw err;
12   var sql = "SELECT * FROM student";
13   con.query(sql, function (err, result, fields) {
14     if (err) throw err;
15     console.log(result[0].rollno+", "+result[0].name);
16   });
17 });
18
```

Ln 18, Col 1 Spaces: 4 UTF-8 LF JavaScript

Output Screen

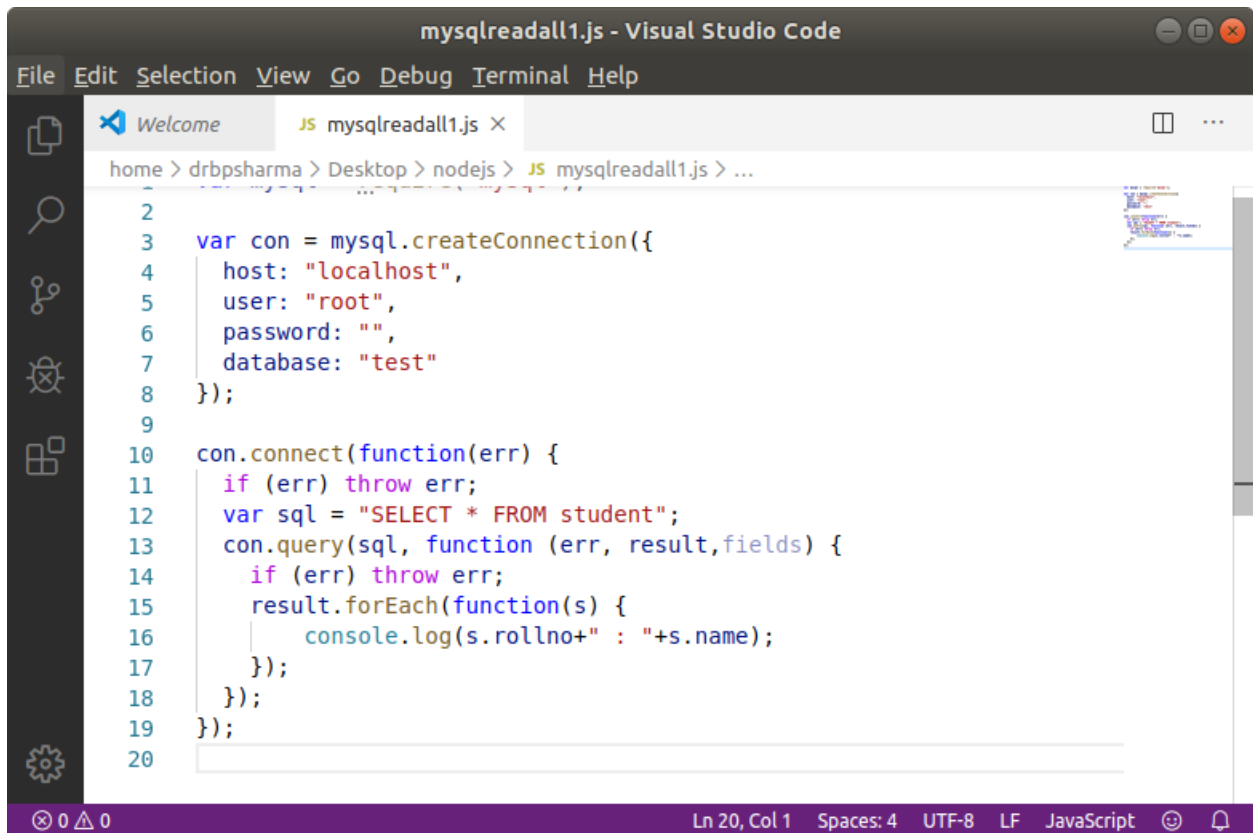


```
drbpsharma@drbpsharma: ~/Desktop/nodejs
File Edit View Search Terminal Help
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node mysqlreadspecificrecord.js
101,Amit Kumar
```

22. How we can view all records from MySQL using NodeJS?

Use `forEach()` loop to read the records one by one.

Try the following code

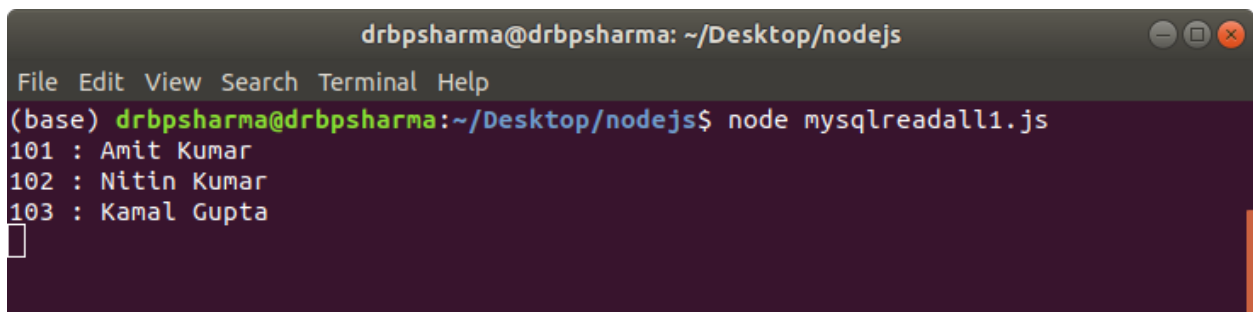


```
mysqlreadall1.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help

Welcome JS mysqlreadall1.js x
home > drbpsharma > Desktop > nodejs > JS mysqlreadall1.js > ...

2
3 var con = mysql.createConnection({
4   host: "localhost",
5   user: "root",
6   password: "",
7   database: "test"
8 });
9
10 con.connect(function(err) {
11   if (err) throw err;
12   var sql = "SELECT * FROM student";
13   con.query(sql, function (err, result, fields) {
14     if (err) throw err;
15     result.forEach(function(s) {
16       console.log(s.rollno+ " : "+s.name);
17     });
18   });
19 });
20
```

Output Screen



```
drbpsharma@drbpsharma: ~/Desktop/nodejs
File Edit View Search Terminal Help
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node mysqlreadall1.js
101 : Amit Kumar
102 : Nitin Kumar
103 : Kamal Gupta
```

23. Can we use ES6 format in NodeJS?

ES6 stands for ECMAScript 6.

ECMAScript was created to standardise JavaScript, and ES6 is the 6th version of ECMAScript, it was published in 2015, and is also known as ECMAScript 2015.

ES6 format is special format for writing classes, arrow functions.

It is recommended in most of new languages like ReactJS, NodeJS etc.

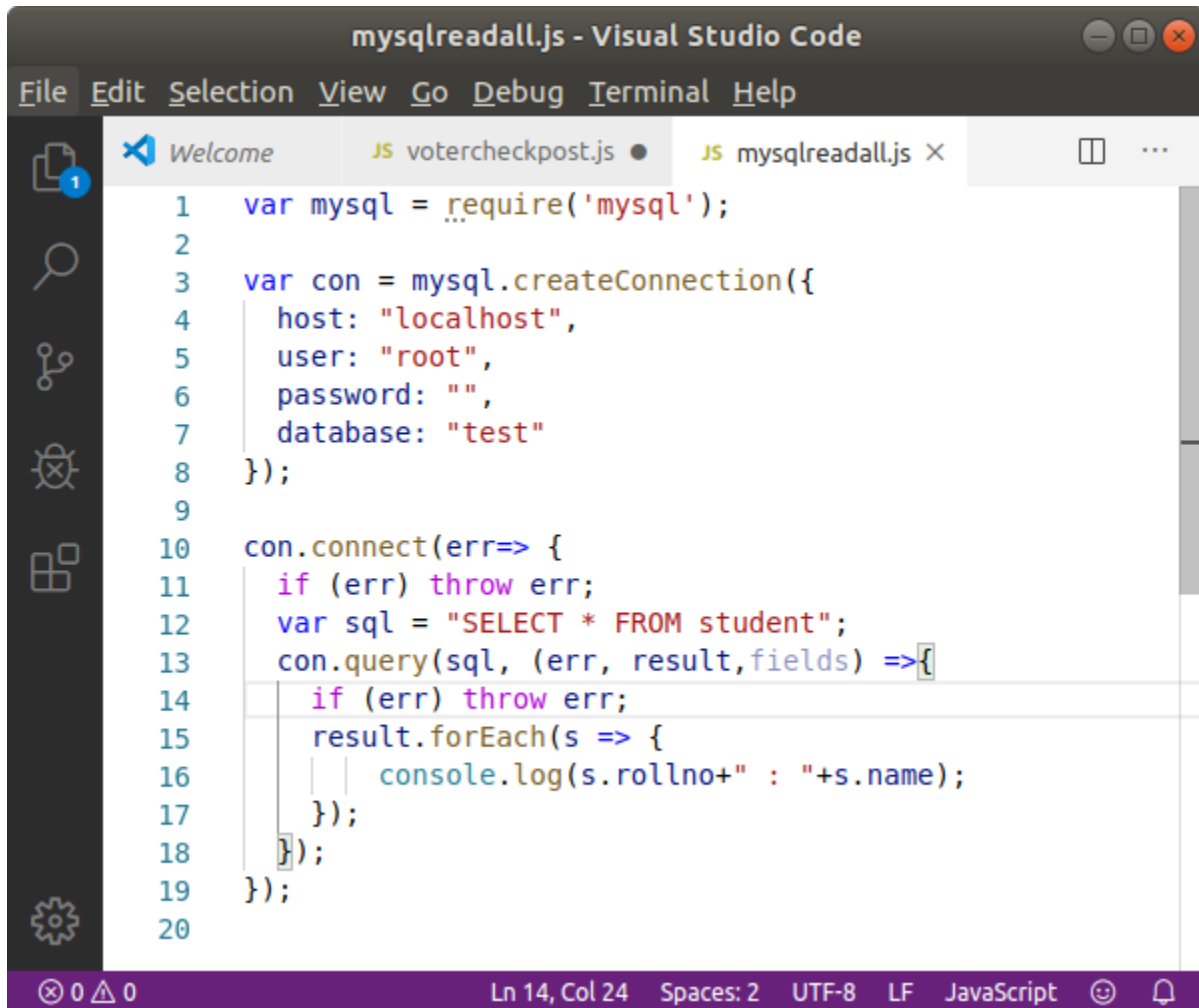
Here we can use the arrow functions without need of function keyword.

Here function(){} means ()=>{}

Here function(x) {} means x=>{}

Here function(a,b,c) {} means (a,b,c)=>{}

Try the following code



```
mysqlreadall.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
Welcome JS votercheckpost.js JS mysqlreadall.js x
1 var mysql = require('mysql');
2
3 var con = mysql.createConnection({
4   host: "localhost",
5   user: "root",
6   password: "",
7   database: "test"
8 });
9
10 con.connect(err=> {
11   if (err) throw err;
12   var sql = "SELECT * FROM student";
13   con.query(sql, (err, result, fields) =>{
14     if (err) throw err;
15     result.forEach(s => {
16       console.log(s.rollno+" : "+s.name);
17     });
18   });
19 });
20
```

Ln 14, Col 24 Spaces: 2 UTF-8 LF JavaScript

Output Screen

```
drbpsharma@drbpsharma: ~/Desktop/nodejs
File Edit View Search Terminal Help
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node mysqlreadall
101 : Amit Kumar
102 : Nitin Kumar
103 : Kamal Gupta
█
```

24. How to work with web requests?

To work with web requests we need to create a web server using methods of [http](#) module.

We need to install the [http](#) module in local folder or global folder

```
npm install http // for local folder
```

```
npm install -g http // for global folder
```

Now we can import the http module into our program using `require()` function

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

Now we can use the `createServer()` function to create a new web server with a callback function with two arguments as request and response.

Call `listen()` function to use the server on some port of your choice.

Use methods of response object to send some data to the web client.

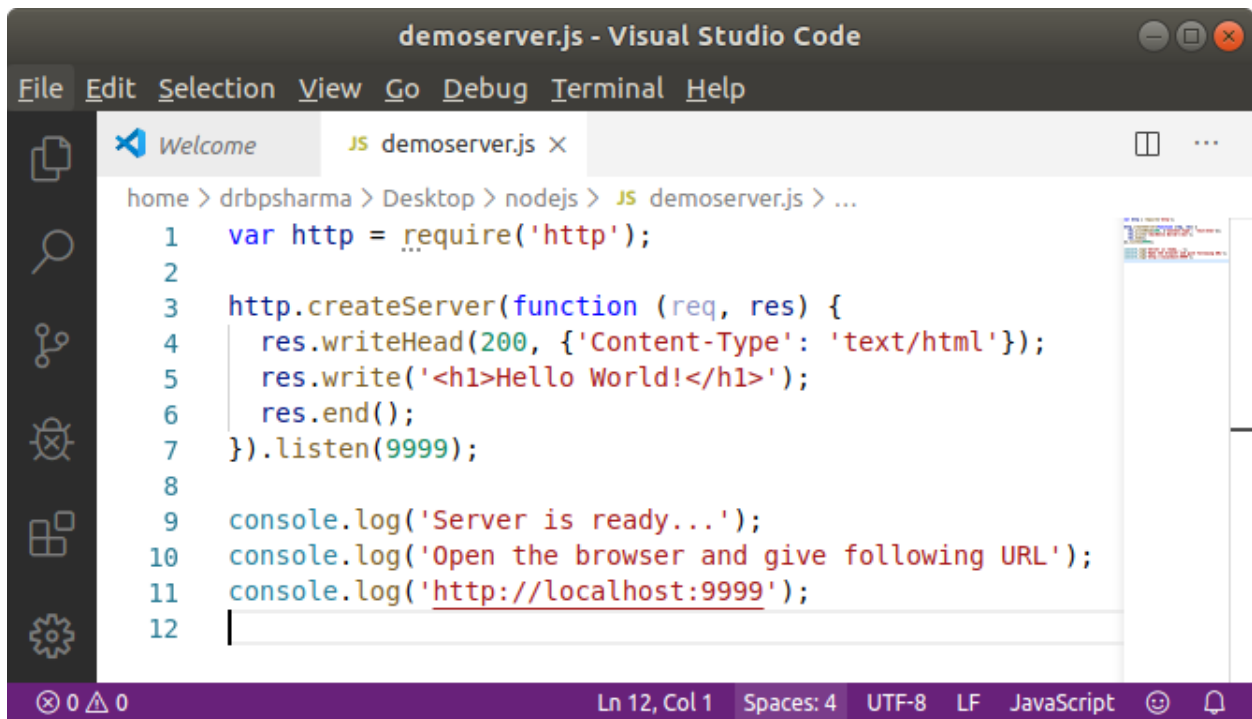
- `write()` -- to write some content on the web page
- `writeHead()` -- to write some meta contents in head section
- `end()` -- to end the response

Example

Create a web application to start the server and say Hello World in Heading style 1.

Try the following steps

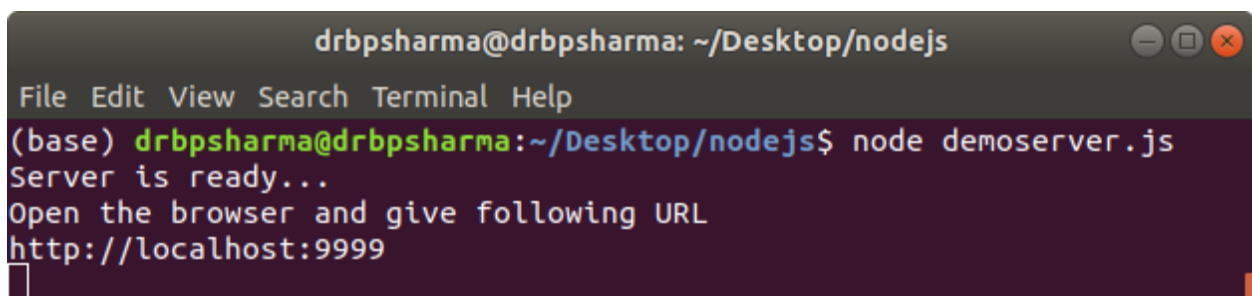
Write the following code in your Visual Studio Code IDE and save as `demoserver.js` and execute it from command prompt



The screenshot shows the Visual Studio Code editor with a file named `demoserver.js` open. The code is as follows:

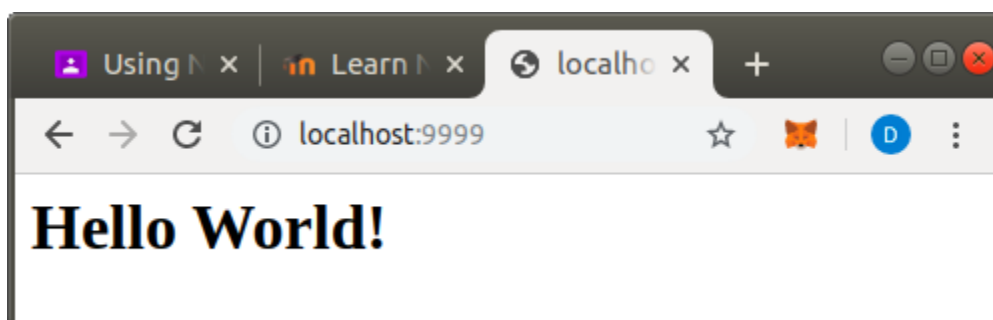
```
1 var http = require('http');
2
3 http.createServer(function (req, res) {
4   res.writeHead(200, {'Content-Type': 'text/html'});
5   res.write('<h1>Hello World!</h1>');
6   res.end();
7 }).listen(9999);
8
9 console.log('Server is ready...');
10 console.log('Open the browser and give following URL');
11 console.log('http://localhost:9999');
12
```

The status bar at the bottom indicates the cursor is at line 12, column 1, with 4 spaces, UTF-8 encoding, LF line endings, and JavaScript language.



The screenshot shows a terminal window with the prompt `drbpsharma@drbpsharma: ~/Desktop/nodejs`. The command `node demoserver.js` has been executed, resulting in the following output:

```
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node demoserver.js
Server is ready...
Open the browser and give following URL
http://localhost:9999
```



25. How to parse the given URL?

When we call some URL using NodeJS, it has different information in it that we can fetch using `parse()` method of `url` module e.g.

- query
- path

- pathname
- host
- hostname
- port

To parse some URL we need to first get the URL from the request argument of the function using url property.

Install the required modules http and url if not installed using npm tool

- npm install http
- npm install url

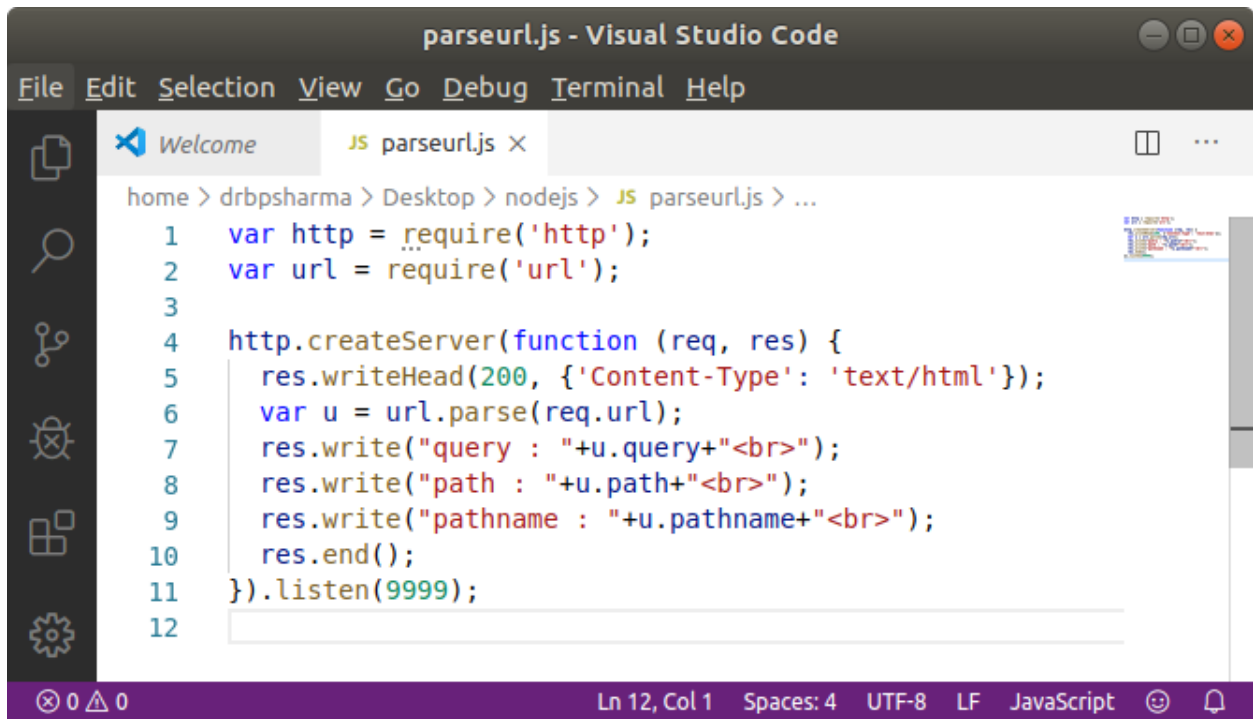
Example

WAP to create a node application which takes some URL and parse that information and show it.

Try following steps

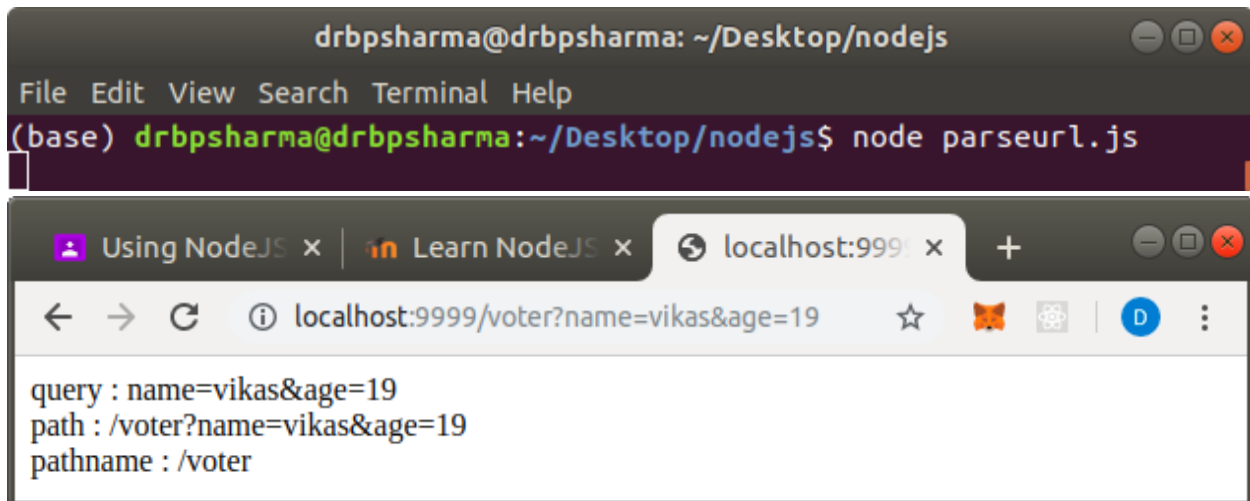
Write the following program code and save it as parseurl.js and execute it using node at the command prompt and give the following url in address bar

http://localhost:9999/voter?name=vikas&age=19



```
parseurl.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help
Welcome JS parseurl.js x
home > drbpsharma > Desktop > nodejs > JS parseurl.js > ...
1 var http = require('http');
2 var url = require('url');
3
4 http.createServer(function (req, res) {
5   res.writeHead(200, {'Content-Type': 'text/html'});
6   var u = url.parse(req.url);
7   res.write("query : "+u.query+"<br>");
8   res.write("path : "+u.path+"<br>");
9   res.write("pathname : "+u.pathname+"<br>");
10  res.end();
11 }).listen(9999);
12
```

Ln 12, Col 1 Spaces: 4 UTF-8 LF JavaScript



26. How to parse data in some query string?

When we pass some data to a web form using ? then called as query string.

For example, if the URL is

`http://localhost:9999/voter?name=Vikas&age=19`

Here the query string is `name=Vikas&age=19`

Using query property of parsed data we can get name and age separately. All data goes in String format.

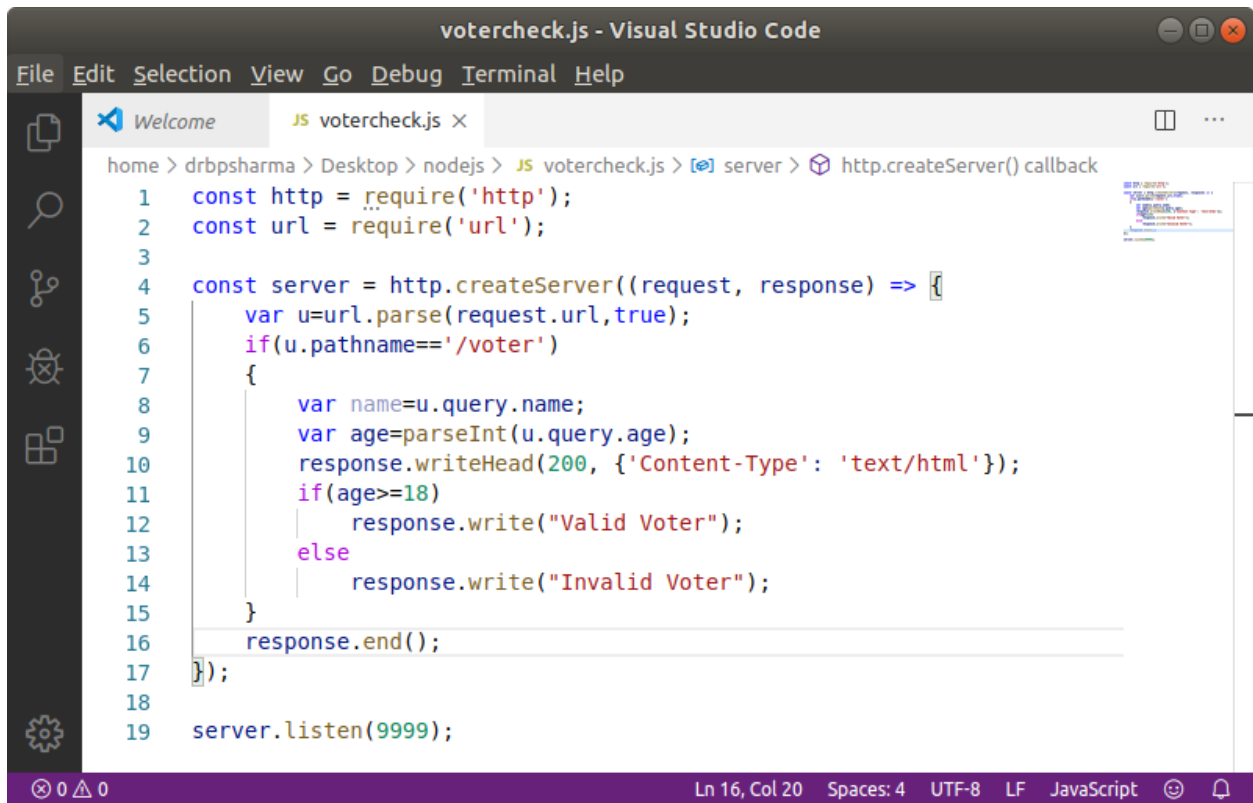
Use `parseInt()` function to convert the string data into integer data.

Example

WAP to input name and age as query string and check whether he/she is eligible for voting or not.

Try following steps

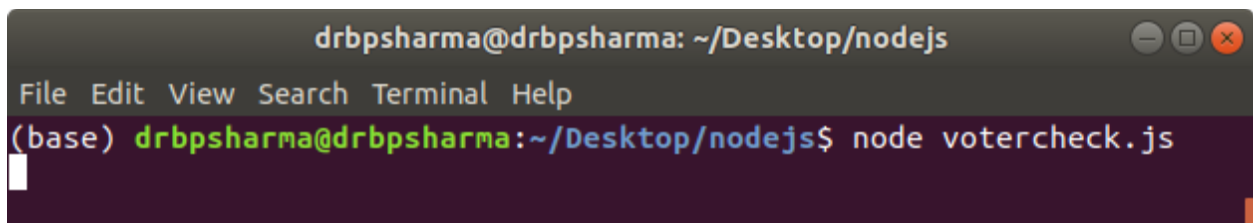
Write the following program code and save file as `votercheck.js` then run it from command prompt.



The screenshot shows the Visual Studio Code editor with a file named `votercheck.js` open. The code is a JavaScript file that uses the `http` module to create a simple web server. It listens on port 9999 and checks if the request path is `/voter`. If the path is correct, it checks the `age` query parameter. If the age is 18 or greater, it returns "Valid Voter"; otherwise, it returns "Invalid Voter".

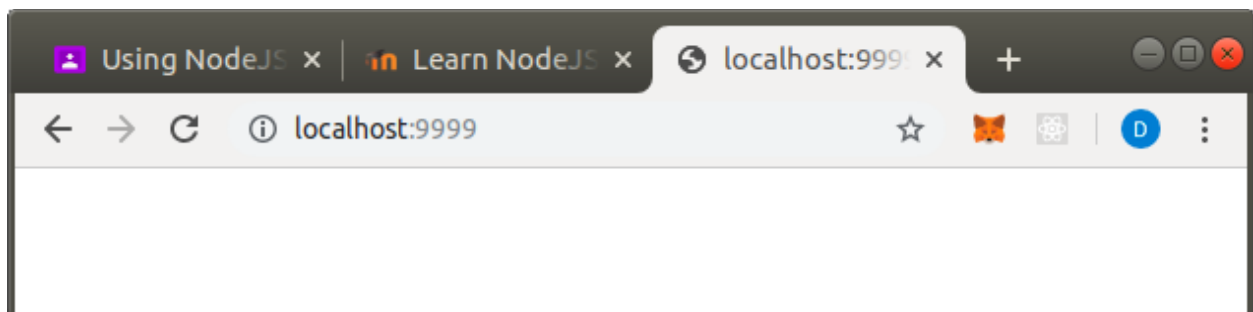
```
1 const http = require('http');
2 const url = require('url');
3
4 const server = http.createServer((request, response) => {
5   var u=url.parse(request.url,true);
6   if(u.pathname=='/voter')
7   {
8     var name=u.query.name;
9     var age=parseInt(u.query.age);
10    response.writeHead(200, {'Content-Type': 'text/html'});
11    if(age>=18)
12      response.write("Valid Voter");
13    else
14      response.write("Invalid Voter");
15  }
16  response.end();
17 });
18
19 server.listen(9999);
```

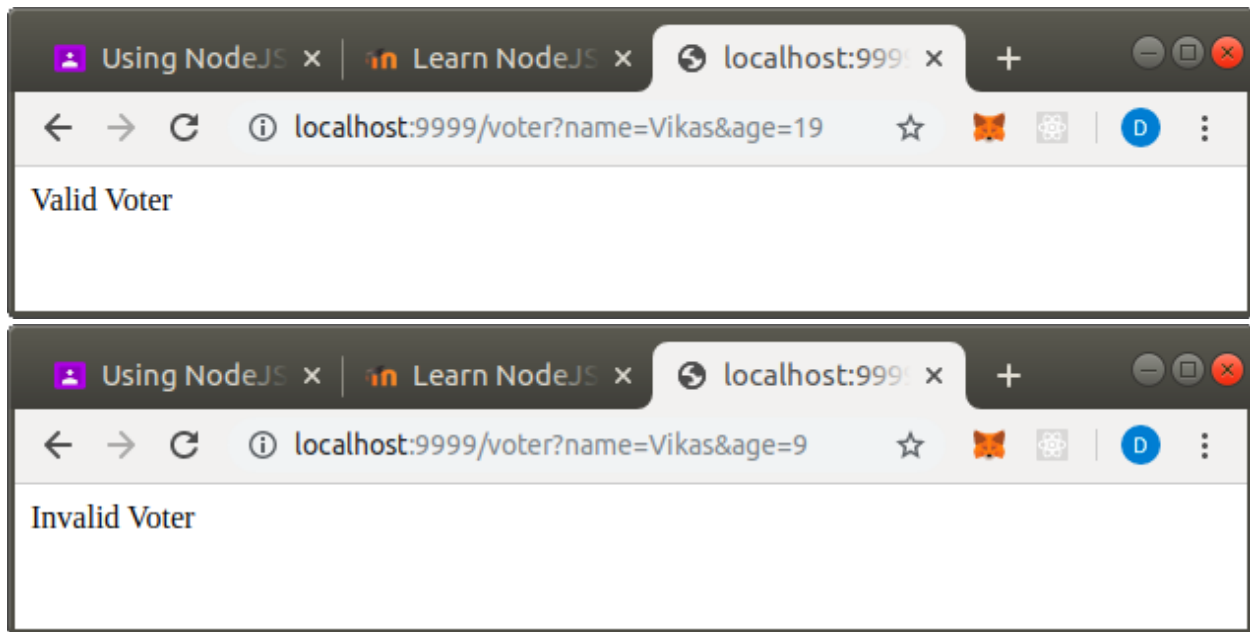
The status bar at the bottom indicates the current position is Line 16, Column 20, with 4 spaces, UTF-8 encoding, LF line endings, and the file is a JavaScript document.



The screenshot shows a terminal window with the prompt `drbpsharma@drbpsharma: ~/Desktop/nodejs`. The command `node votercheck.js` has been executed, and the terminal is now ready for further input.

Now open the browser and give the following URLs and test the results





27. How to submit data using a web form?

Install and use the formidable module of NodeJS to parse the data submitted by the form.

First install the module from command prompt

```
npm install formidable
```

Import the module in your program

```
var formidable=require('formidable');
```

Create an instance of the IncomingForm() class from formidable module

```
var form=new formidable.IncomingForm();
```

Now use the parse() method of form reference with request to be parsed and a callback to handle the parsed information

```
form.parse(request, function(err,fields){});
```

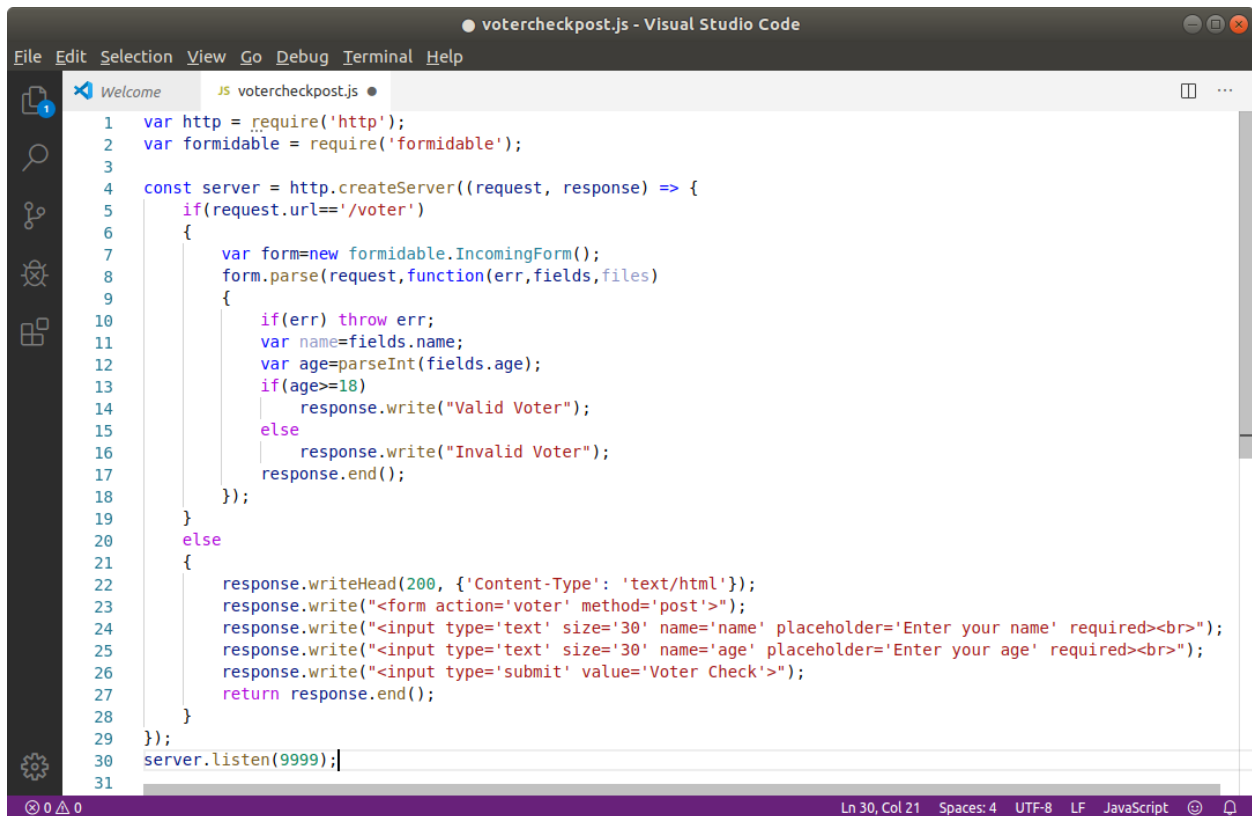
The fields argument receives all the form fields submitted by the form.

Example

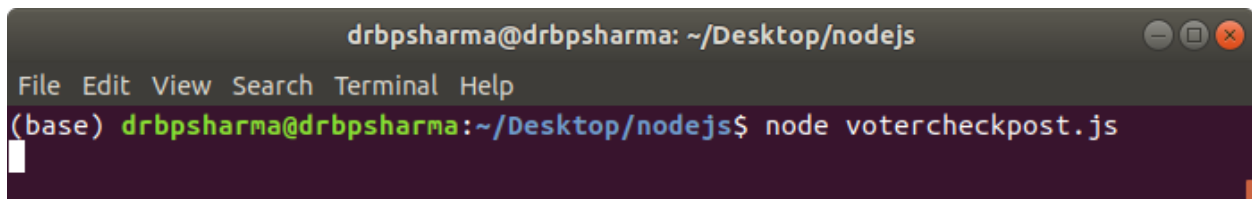
Write a program to input name and age of a person from user using post method and check it to be valid voter.

Try following steps

Write the following code in Visual Studio Code IDE, save file as logincheckpost.js and run it from command prompt

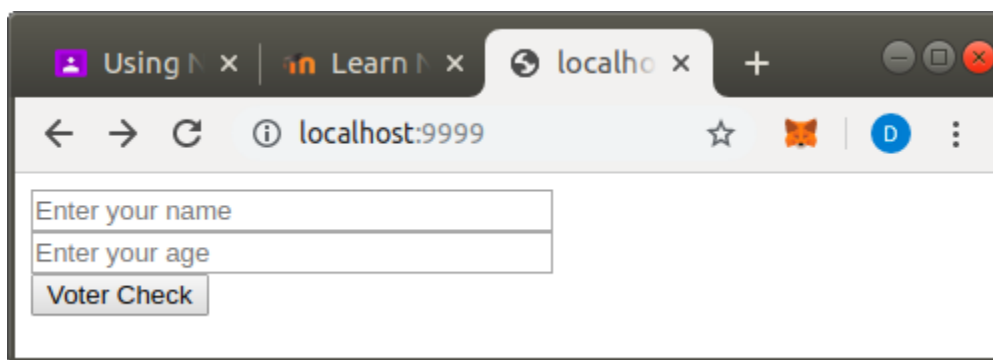


```
1 var http = require('http');
2 var formidable = require('formidable');
3
4 const server = http.createServer((request, response) => {
5   if(request.url== '/voter')
6   {
7     var form=new formidable.IncomingForm();
8     form.parse(request,function(err,fields,files)
9     {
10      if(err) throw err;
11      var name=fields.name;
12      var age=parseInt(fields.age);
13      if(age>=18)
14        response.write("Valid Voter");
15      else
16        response.write("Invalid Voter");
17      response.end();
18    });
19  }
20  else
21  {
22    response.writeHead(200, {'Content-Type': 'text/html'});
23    response.write("<form action='voter' method='post'>");
24    response.write("<input type='text' size='30' name='name' placeholder='Enter your name' required><br>");
25    response.write("<input type='text' size='30' name='age' placeholder='Enter your age' required><br>");
26    response.write("<input type='submit' value='Voter Check'>");
27    return response.end();
28  }
29 });
30 server.listen(9999);
31
```



```
drbpsharma@drbpsharma: ~/Desktop/nodejs
File Edit View Search Terminal Help
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node votercheckpost.js
```

Now open the web browser and give the following URL



28. Assignment

Write a program to input roll number and name from user using a web form and save into student table of test database in MySQL as we did earlier using NodeJS Framework.

29. How to upload a file using NodeJS?

To upload a file you need to define the file control in your web form to select a file with enctype attribute.

```
<input type='file' enctype='multipart/form-data'>
```

Give some name to the control e.g. `<input type='file' name='photo'>`

When this form field goes to the NodeJS server, we need to parse it use `parse()` method of formidable module using third argument of callback called files.

The file by default uploaded in temporary folder with temporary name that we can get using `path` property and actual file name using `name` property.

Example

Create a web form to upload the photos to create a photo gallery with photo, title and description. Upload the file in folder of your choice and save file information into test database of MySQL database having gallery table. Every photo must have a unique auto generated photoid.

Try following steps

Start the MySQL server and create a table gallery in test database with following columns

```
Create table gallery(photoid int primary key auto_increment, title varchar(100), description  
varchar(100), photofile varchar(100));
```

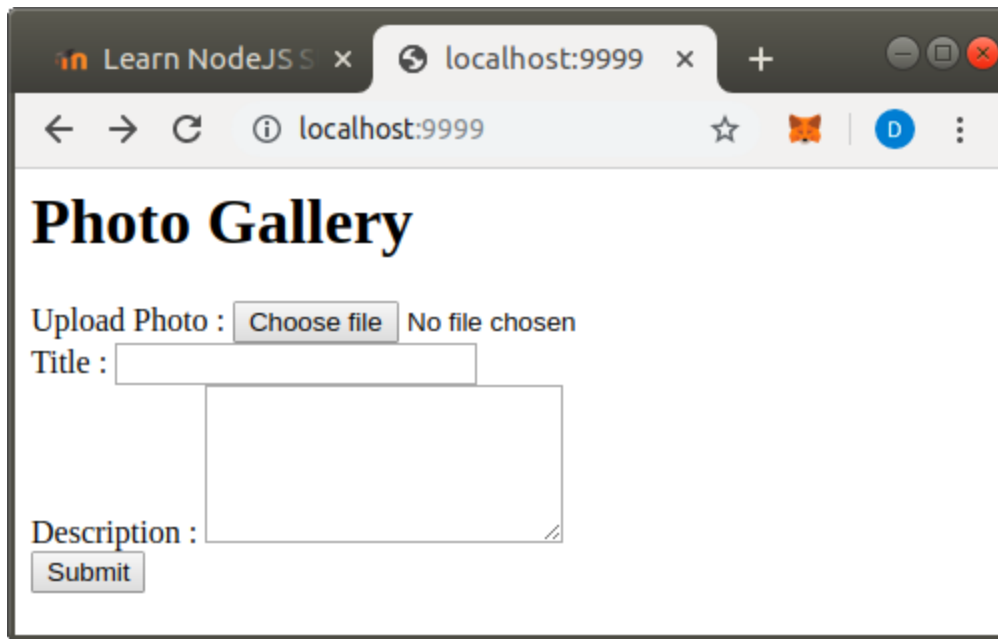
Now write the following code and save the file as `fileupload.js` and run it on command prompt

```
fileupload.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help

Welcome  js fileupload.js x

1  var http = require('http');
2  var formidable = require('formidable');
3  var fs = require('fs');
4  var mysql = require('mysql');
5
6  http.createServer(function (req, res) {
7    if (req.url == '/fileupload') {
8      var form = new formidable.IncomingForm();
9      form.parse(req, function (err, fields, files) {
10        var title=fields.title;
11        var description=fields.description;
12        var filename=files.photo.name;
13
14        var con = mysql.createConnection({host: "localhost",user: "root",password: "",database: "test",port:"3306"});
15        con.connect(function(err) {
16          if (err) throw err;
17          var sql = "INSERT INTO gallery (title, description,photofile) VALUES ('"+title+"','"+description+"','"+filename+"')";
18          con.query(sql, function (err, result) {
19            if (err) throw err;
20            console.log("Record saved");
21          });
22        });
23
24        var oldpath = files.photo.path;
25        var newpath = '/home/drpbsharma/Desktop/nodejs/uploads/' + files.photo.name;
26        fs.rename(oldpath, newpath, function (err) {
27          if (err) throw err;
28          res.write('File uploaded and moved!');
29          res.end();
30        });
31      });
32    }
33    else
34    {
35      res.writeHead(200, {'Content-Type': 'text/html'});
36      res.write('<h1>Photo Gallery</h1>');
37      res.write('<form action="fileupload" method="post" enctype="multipart/form-data">');
38      res.write('Upload Photo : <input type="file" name="photo"><br>');
39      res.write('Title : <input type="text" name="title"><br>');
40      res.write('Description : <textarea name="description" rows="5" cols="20"></textarea><br>');
41      res.write('<input type="submit">');
42      res.write('</form>');
43      return res.end();
44    }
45  }).listen(9999);
46
```

```
drbpsharma@drbpsharma: ~/Desktop/nodejs
File Edit View Search Terminal Help
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node fileupload.js
Record saved
█
```



30. How to use the MongoDB with NodeJS?

MongoDB is a No SQL database which do not have any table structure and do not require SQL statements.

You need to install MongoDB on your machine depending on the operating system.

For Linux, use the steps from given link

<https://www.digitalocean.com/community/tutorials/how-to-install-mongodb-on-ubuntu-18-04>

For Windows, use the steps in given link

<https://docs.mongodb.com/manual/tutorial/install-mongodb-on-windows/>

MongoDB server by default installs on port number 27017 just like MySQL which installs on 3306

Now need to install the mongodb package using NPM

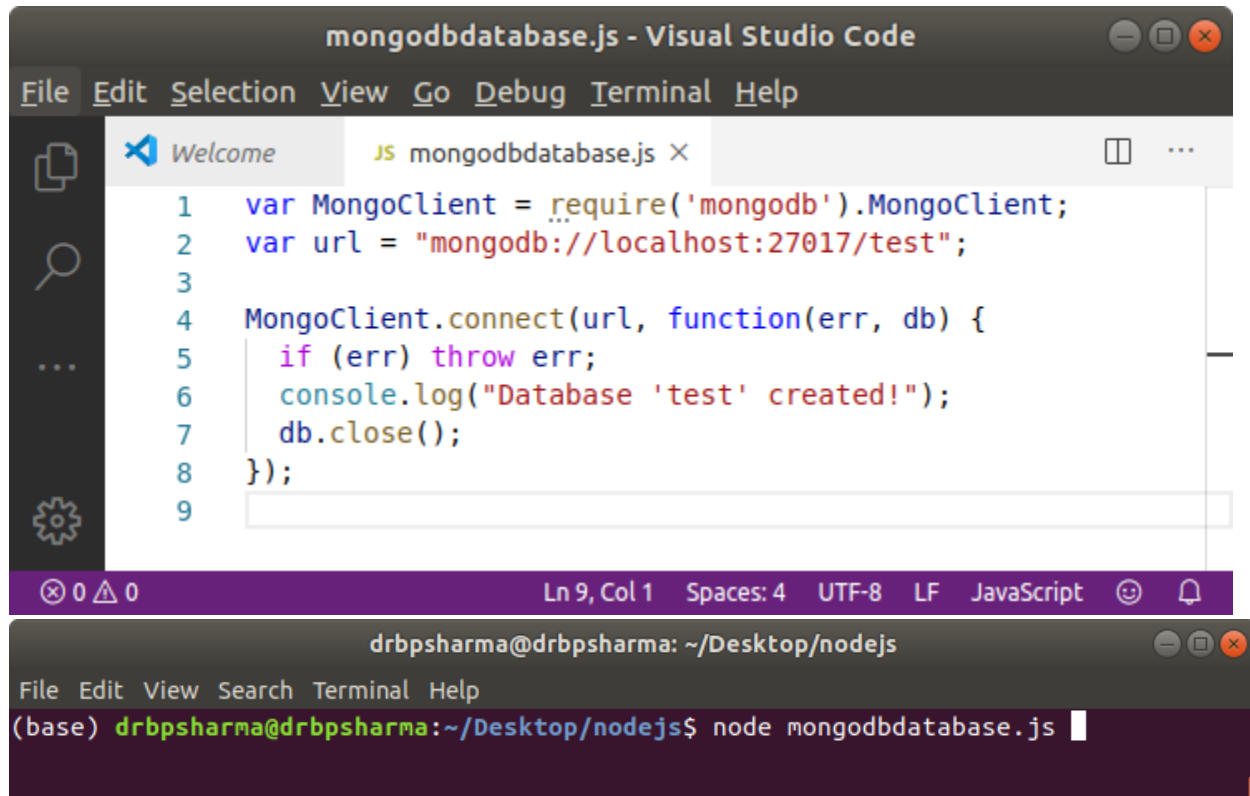
```
npm install mongodb
```

Now you can import the mongodb module in your program

```
var mongo = require('mongodb');
```


31. How to create a database in MongoDB?

Write the following code, save as `mongodbdatabase.js` and run it to create a database named as `test`



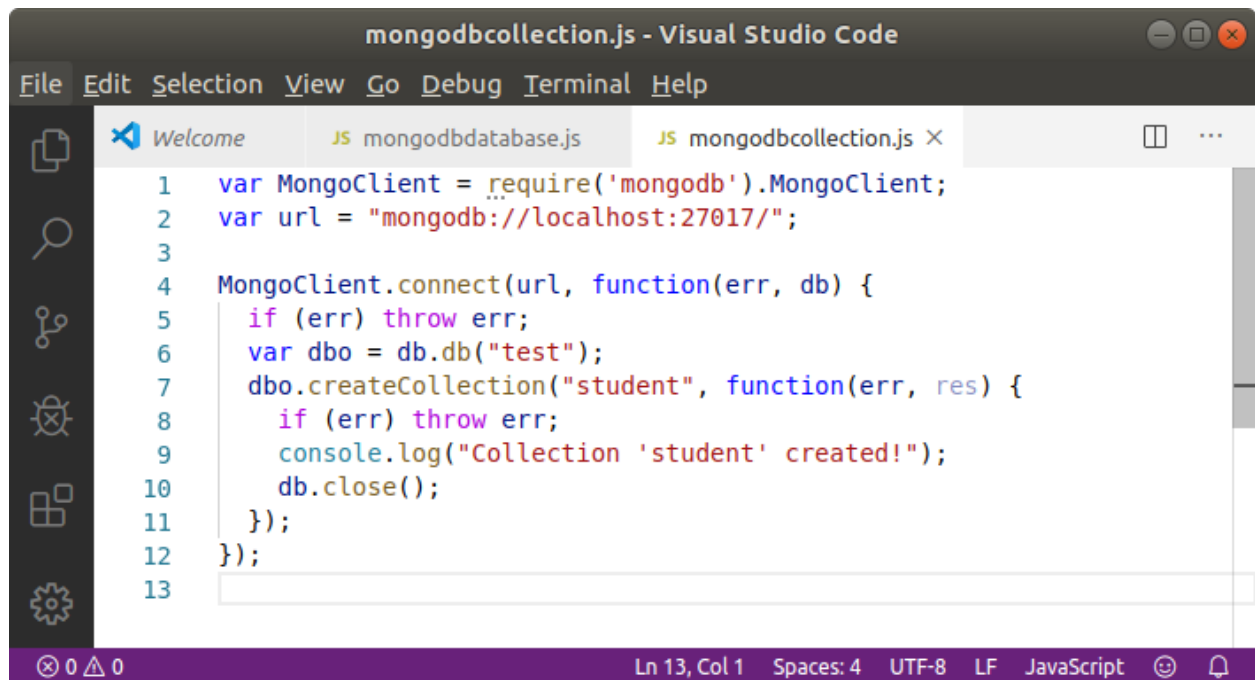
The image shows a Visual Studio Code editor window titled "mongodbdatabase.js - Visual Studio Code" with a menu bar (File, Edit, Selection, View, Go, Debug, Terminal, Help) and a sidebar. The editor displays the following JavaScript code:

```
1 var MongoClient = require('mongodb').MongoClient;
2 var url = "mongodb://localhost:27017/test";
3
4 MongoClient.connect(url, function(err, db) {
5   if (err) throw err;
6   console.log("Database 'test' created!");
7   db.close();
8 });
9
```

Below the editor is a terminal window with the title "drbpsharma@drbpsharma: ~/Desktop/nodejs". The terminal shows the command `node mongodbdatabase.js` being executed.

32. How to create a collection in MongoDB?

MongoDB do not have the tables but they have the collections. Data get managed very similar to dictionary collection in python using key/value pairs

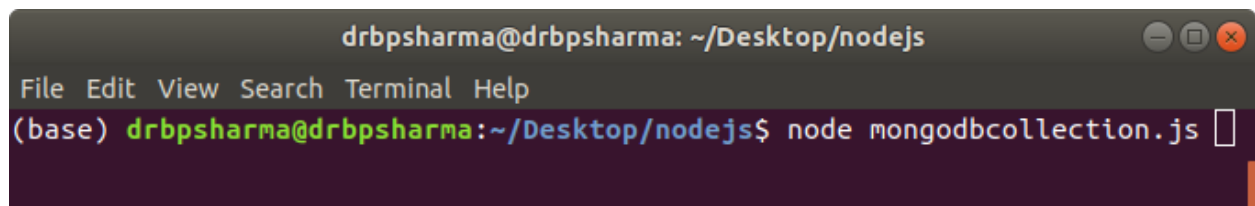


```
mongodbcollection.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help

Welcome JS mongodbdatabase.js JS mongodbcollection.js X

1 var MongoClient = require('mongodb').MongoClient;
2 var url = "mongodb://localhost:27017/";
3
4 MongoClient.connect(url, function(err, db) {
5   if (err) throw err;
6   var dbo = db.db("test");
7   dbo.createCollection("student", function(err, res) {
8     if (err) throw err;
9     console.log("Collection 'student' created!");
10    db.close();
11  });
12 });
13
```

Ln 13, Col 1 Spaces: 4 UTF-8 LF JavaScript

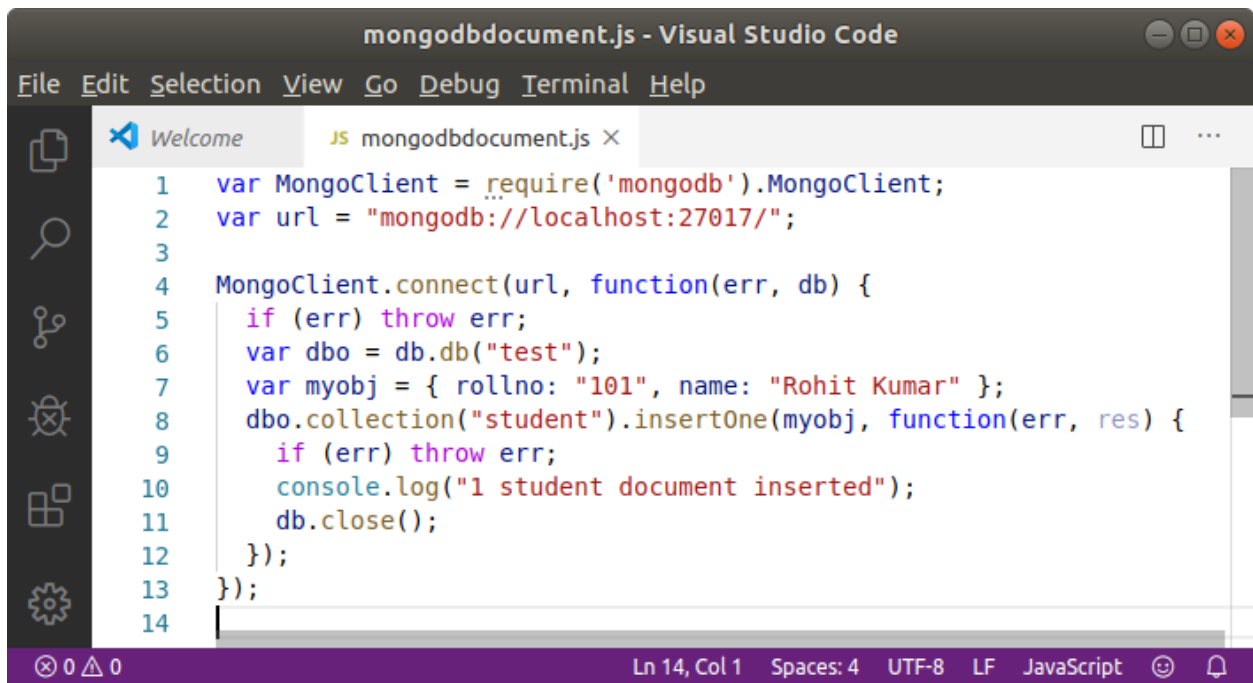


```
drbpsharma@drbpsharma: ~/Desktop/nodejs
File Edit View Search Terminal Help
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node mongodbcollection.js
```

33. How to insert a document into MongoDB?

MongoDB uses the term document for a record. Every document is some JSON object.

Use insertOne() method to pass a JSON record to save into collection

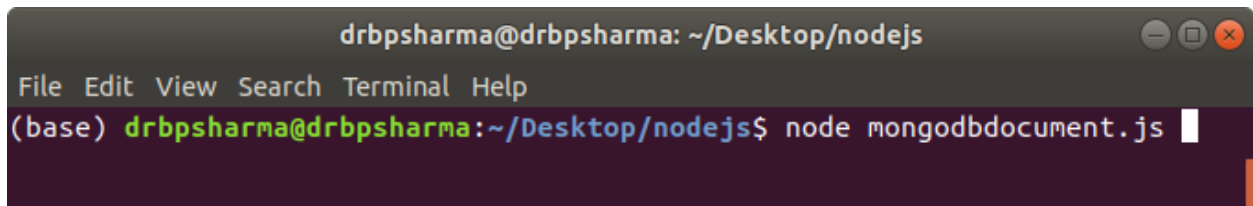


```
mongodbdocument.js - Visual Studio Code
File Edit Selection View Go Debug Terminal Help

Welcome JS mongodbdocument.js x

1 var MongoClient = require('mongodb').MongoClient;
2 var url = "mongodb://localhost:27017/";
3
4 MongoClient.connect(url, function(err, db) {
5   if (err) throw err;
6   var dbo = db.db("test");
7   var myobj = { rollno: "101", name: "Rohit Kumar" };
8   dbo.collection("student").insertOne(myobj, function(err, res) {
9     if (err) throw err;
10    console.log("1 student document inserted");
11    db.close();
12  });
13 });
14
```

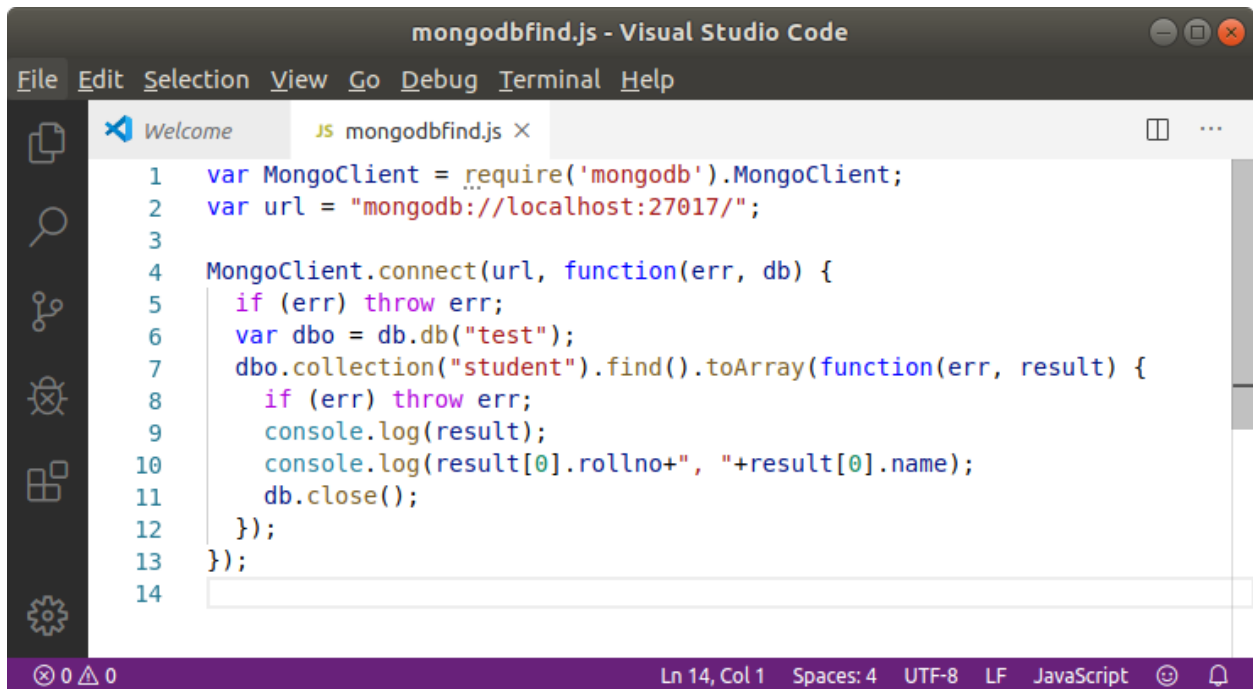
Ln 14, Col 1 Spaces: 4 UTF-8 LF JavaScript



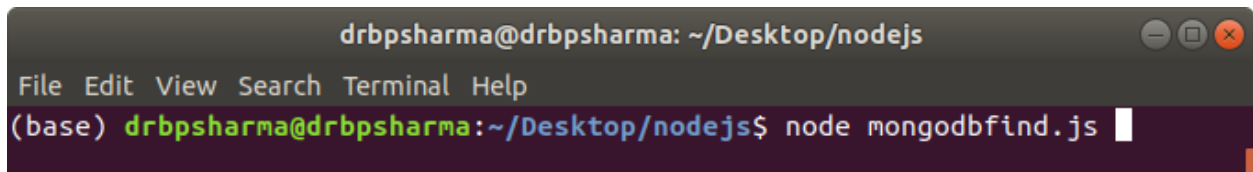
```
drbpsharma@drbpsharma: ~/Desktop/nodejs
File Edit View Search Terminal Help
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node mongodbdocument.js
```

34. How to view documents in MongoDB collection?

Use `find()` function to fetch the documents. Convert them to array using `toArray()` method for easy access.



```
1 var MongoClient = require('mongodb').MongoClient;
2 var url = "mongodb://localhost:27017/";
3
4 MongoClient.connect(url, function(err, db) {
5   if (err) throw err;
6   var dbo = db.db("test");
7   dbo.collection("student").find().toArray(function(err, result) {
8     if (err) throw err;
9     console.log(result);
10    console.log(result[0].rollno+" "+result[0].name);
11    db.close();
12  });
13 });
14
```



```
drbpsharma@drbpsharma: ~/Desktop/nodejs
File Edit View Search Terminal Help
(base) drbpsharma@drbpsharma:~/Desktop/nodejs$ node mongodbfnd.js
```

35. How to import data in JSON file to MongoDB?

We can directly import any data in JSON file into MongoDB collection using mongoimport command

Suppose you have a file as users.json in current folder having data as given below

```
{
  "email": "bpsharma@gmail.com",
  "admin": true,
  "password": "123456"
}
```

If you want to import the data into MongoDB server into users collection of test database then issue the following command on command prompt

```
mongoimport --db test --collection users --file users.json --jsonArray
```

36. How we can export data from MongoDB collection to some JSON file?

To export data from a collection in MongoDB into some file use the following command

```
mongoexport --db test --collection users > users.json
```

Here test is the database, users is the collection and data will be saved into users.json file

37. How to encode JSON to Base64?

First convert the JSON data into string and then convert the data to single base64 string format using the given sample code

Example 1

```
var b64 = Buffer.from(JSON.stringify({"hello":"world"})).toString("base64")  
  
console.log(b64);
```

Example 2

```
var data={"hello":"world"}  
  
var b64 = Buffer.from(JSON.stringify(data)).toString("base64")  
  
console.log(b64);
```

38. How to decode Base64 to JSON?

Use the following code to decode the Base64 format into JSON

Example

```
var data="eyJhZGUiOiJhZGUi";  
  
var b64 = Buffer.from(data,"base64").toString("ascii")  
  
console.log(b64);
```