**Introduction to Node.Js**

Node.js is an open-source and cross-platform JavaScript runtime environment. It is a popular tool for almost any kind of project!

Node.js runs the V8 JavaScript engine, the core of Google Chrome, outside of the browser. This allows Node.js to be very performant.

A Node.js app runs in a single process, without creating a new thread for every request. Node.js provides a set of asynchronous I/O primitives in its standard library that prevent JavaScript code from blocking and generally, libraries in Node.js are written using non-blocking paradigms, making blocking behavior the exception rather than the norm.

When Node.js performs an I/O operation, like reading from the network, accessing a database or the filesystem, instead of blocking the thread and wasting CPU cycles waiting, Node.js will resume the operations when the response comes back.

This allows Node.js to handle thousands of concurrent connections with a single server without introducing the burden of managing thread concurrency, which could be a significant source of bugs.

Node.js has a unique advantage because millions of frontend developers that write JavaScript for the browser are now able to write the server-side code in addition to the client-side code without the need to learn a completely different language.

**Advantage of Node.js**

1. Robust technology stack. ...

2. Fast-processing and event-based model. ...

3. Scalable technology for microservices. ...

4. Rich ecosystem. ...

5. Strong corporate support. ...

6. Seamless JSON support. ...

7. Performance bottlenecks with heavy computation tasks. ...

8. Callback hell issue.

**Disadvantage of Node.js**

### 1. Application Programming Interface (API) is Not Stable

### 2. Does not have a Strong Library Support System

### 3. Asynchronous Programming Model

4. Reduces performance when handling Heavy Computing Tasks.

**Installation of Node.Js**

**Step 1: Download Node.js Installer**

In a web browser, navigate to <https://nodejs.org/en/download/>. Click the Windows Installer button to download the latest default version. At the time this article was written, version 10.16.0-x64 was the latest version. The Node.js installer includes the NPM package manager.

**Step 2: Install Node.js and NPM from Browser**

1. Once the installer finishes downloading, launch it. Open the downloads link in your browser and click the file. Or, browse to the location where you have saved the file and double-click it to launch.

2. The system will ask if you want to run the software – click Run.

3. You will be welcomed to the Node.js Setup Wizard – click Next.

4. On the next screen, review the license agreement. Click Next if you agree to the terms and install the software.

5. The installer will prompt you for the installation location. Leave the default location, unless you have a specific need to install it somewhere else – then click Next.

6. The wizard will let you select components to include or remove from the installation. Again, unless you have a specific need, accept the defaults by clicking Next.

7. Finally, click the Install button to run the installer. When it finishes, click Finish.

**PRACTICAL NO. 1**

**CREATE AN APPLICATION TO DEMONSTRATE NODE.JS MODULES**

**PRACTICAL NO. 1**

**CREATE AN APPLICATION TO DEMONSTRATE NODE.JS MODULES**

**Q. 1 User Define Module**

**Program:**

**module\_ext2.js**

exports.addition=function add(a,b)

{

return a+b;

}

exports.subtraction=function add(a,b)

{

return a-b;

}

exports.multiplication=function add(a,b)

{

return a\*b;

}

exports.division=function add(a,b)

{

return a/b;

}

exports.modulus=function add(a,b)

{

return a%b;

}

**appcal.js**

var calc=require('./module\_ext2');

res=calc.addition(10,20);

subtraction=calc.subtraction(10,20);

multiplication=calc.multiplication(10,20);

division=calc.division(10,20);

modulus=calc.modulus(10,20);

console.log("The Addition Is: "+res);

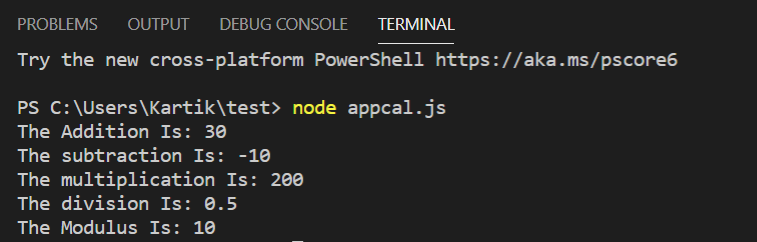
console.log("The subtraction Is: "+subtraction);

console.log("The multiplication Is: "+multiplication);

console.log("The division Is: "+division);

console.log("The Modulus Is: "+modulus);

**Output :**

****

**Q.2 Find the area of rectangle,Circle and square using modules.**

**area\_of\_rectangle.js**

exports.rectangle=function rect(a,b)

{

return a\*b;

}

exports.circle=function cir(a)

{

return Math.PI\*a\*a;

}

exports.square=function squ(a)

{

return a\*a;

}

**Calculation.js**

const prompt = require('prompt-sync')();

var a = parseInt(prompt('Enter the first number: '));

var b = parseInt(prompt('Enter the second number: '));

var calc=require('./area\_of\_rectangle');

rect=calc.rectangle(a,b);

cir=calc.circle(a);

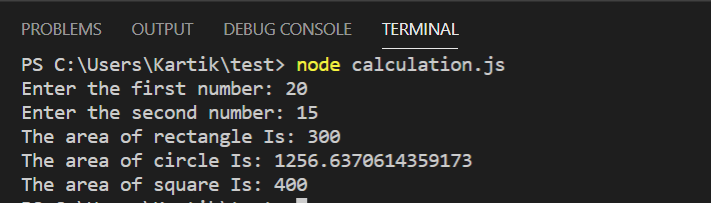
squ=calc.square(a);

console.log("The area of rectangle Is: "+rect);

console.log("The area of circle Is: "+cir);

console.log("The area of square Is: "+squ);

**Output :**

****

**PRACTICAL NO. 2**

**CREATE AN APPLICATION TO DEMONSTRATE VARIOUS NODE.JS EVENTS**

**PRACTICAL NO. 2**

**CREATE AN APPLICATION TO DEMONSTRATE VARIOUS NODE.JS**

**EVENTS**

**Q. 1 Event Listners**

**Program:**

**event.js**

const EventEmitter=require('events');

const emitter=new EventEmitter();

//regstring a listener

emitter.on('msgname',function()

{

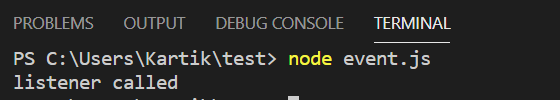
console.log('listener called');

});

//rise an event

emitter.emit('msgname');

**Output :**

****

**event1.js**

const EventEmitter=require('events');

const emitter=new EventEmitter();

emitter.on('msgname',(pageno,msg) =>

{

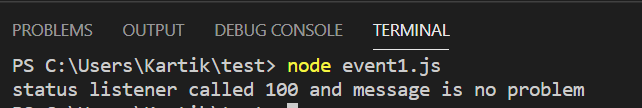
console.log(`status listener called ${pageno} and message is ${msg}`);

});

//rise an event

emitter.emit('msgname',100,"no problem");

**Output :**

****

**Eventemitter.js**

var emitter = require('events').EventEmitter;

function LoopProcessor(num) {

var e = new emitter();

setTimeout(function () {

for (var i = 1; i <= num; i++) {

e.emit('BeforeProcess', i);

console.log('Processing number:' + i);

e.emit('AfterProcess', i);

}

}

, 2000);

return e;

}

var lp = LoopProcessor(3);

lp.on('BeforeProcess', function (data) {

console.log('About to start the process for ' + data);

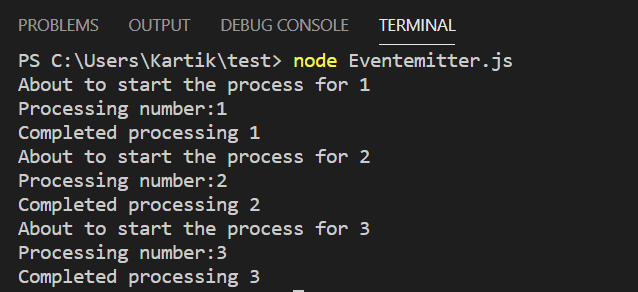
});

lp.on('AfterProcess', function (data) {

console.log('Completed processing ' + data);

});

**Output :**

****

**eventEmmiter.js**

const events =require("events");

const eventEmitter=new events.EventEmitter();

function listener1()

{

console.log("event recived by listener1");

}

function listener2()

{

console.log("event recived by listener2");

}

function listener3()

{

console.log("event recived by listener3");

}

eventEmitter.addListener("write",listener1);

eventEmitter.on("write",listener2);

eventEmitter.on("write",listener3);

eventEmitter.emit("write");

console.log(eventEmitter.listenerCount("write"));

eventEmitter.removeListener("write",listener1);

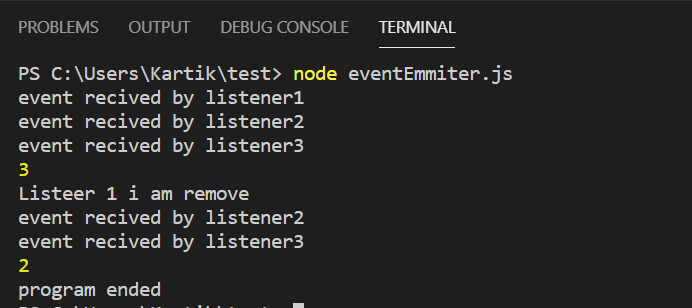
console.log("Listeer 1 i am remove");

eventEmitter.emit("write");

console.log(eventEmitter.listenerCount("write"));

console.log("program ended");

**Output :**

****

**PRACTICAL NO. 3**

**CREATE AN APPLICATION TO DEMONSTRATE NODE.JS FUNCTIONS**

**PRACTICAL NO. 3**

**CREATE AN APPLICATION TO DEMONSTRATE NODE.JS FUNCTIONS**

**Q. 1 Callback Function**

**Program:**

**callback.js**

const mess=function()

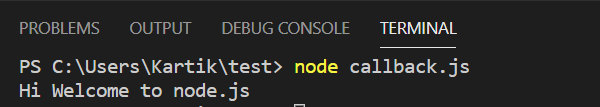
{

console.log("Hi Welcome to node.js");

}

setTimeout(mess,3000);

**Output :**

****

**Q. 2 Arrow Callback Function**

**callback.js**

const mess=function()

{

console.log("Hi Welcome to node.js");

}

setTimeout(mess,3000);

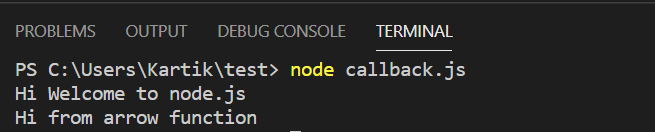
//arrow call back function

setTimeout(() => {

console.log("Hi from arrow function");

}, 4000);

**Output :**

****

**Q.3 Delegate function with callback**

function display(params)

{

console.log(params);

// this is form deligate function

// part of call back function

}

function calc(a,b,mycal)

{

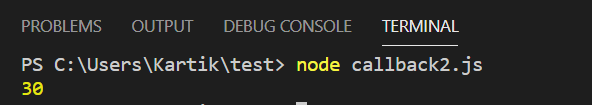
var sum=a+b;

mycal(sum);

}

calc(10,20,display);

**Output :**

****

**Q.4 Calculator**

**calculator.js**

const prompt=require('prompt-sync')();

var a=parseInt(prompt('Enter first number: '));

var b=parseInt(prompt('Enter second number: '));

var c;

function add(a,b)

{

c=a+b;

console.log(`Addition is: `,+c);

}

var r=add(a,b);

function sub(a,b)

{

c=a-b;

console.log(`Subtraction is: `,+c);

}

var r=sub(a,b);

function mul(a,b)

{

c=a\*b;

console.log(`Multiplication is: ${c}`);

}

var r=mul(a,b);

function div(a,b)

{

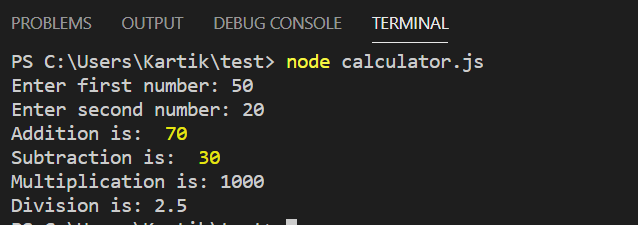
c=a/b;

console.log(`Division is: ${c}`);

}

var r=div(a,b);

**Output :**

****

**Q.5 Fibonacci series**

**function.js**

const prompt=require('prompt-sync')();

function fibonacci()

{

var num1=0;

var num2=1;

var num3;

var cout;

var i;

cout=parseInt(prompt('Enter a Number: '));

console.log(`${num1}`);

console.log(`${num2}`);

for(i=2;i<cout;i++)

{

num3=num1+num2;

console.log(`${num3}`);

num1=num2;

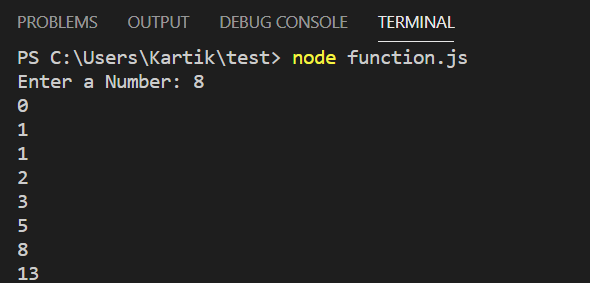
num2=num3;

}

}

var r=fibonacci();

**Output :**

****

**Q. 6 SetTimer: without param**

**Write a program to display “Hi,Good Morning” after each 1 second till 6 times after that clear time to initial state and print “Done Displaying”.**

**settimedemo.js**

var cnt=0;

const myfun=()=>

{

console.log("Hi,Good Morning");

cnt=cnt+1;

if(cnt==6)

{

console.log("Done");

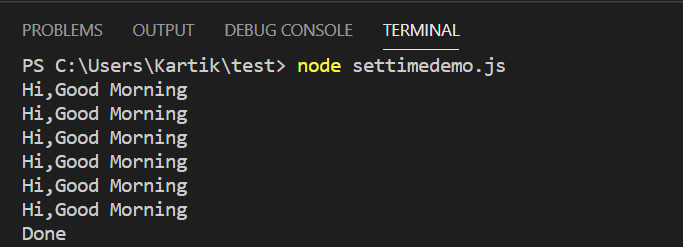
clearInterval(timerId);

}

};

const timerId=setInterval(myfun,1000);

**Output :**



**Q. 7 SetTimer: without function pram**

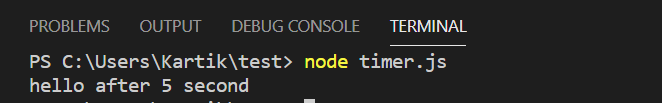
**timer.js**

setTimeout(() => {

console.log("hello after 5 second");

}, 5000);

**Output :**



**Q. 8 SetTimer: with Parameter**

**timer\_using\_parameter.js**

const myfun=(p1)=>

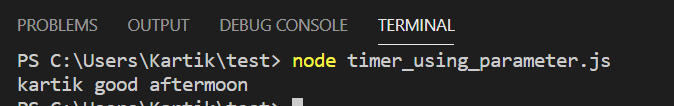
{

console.log(p1,"good aftermoon");

};

setTimeout(myfun,3000,'kartik');

**Output :**

****

**Q.9 SetTimer: With Multi Parameter**

**multi\_para\_timer.js**

const myfun=(p1)=>

{

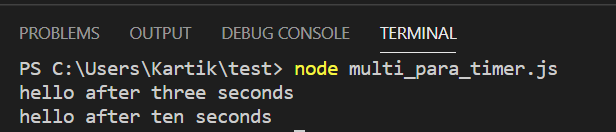
console.log('hello after '+p1+' seconds');

};

setTimeout(myfun,5000,"three");

setTimeout(myfun,10\*1000,"ten");

**Output :**

****

**Q.10 SetTimer: Display 2 message after 5 millisecond and 10millisecond**

**multi\_para\_timer.js**

const myfun=(p1)=>

{

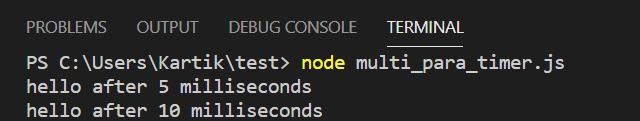
console.log('hello after '+p1+' milliseconds');

};

setTimeout(myfun,500,"5");

setTimeout(myfun,1000,"10");

**Output:**

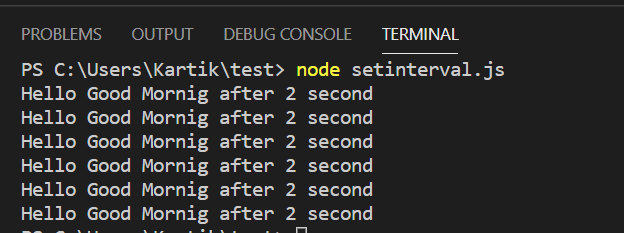
****

**Q.11 SetInterval: it wil delay operation in loop**

**setinterval.js**

setInterval(() => console.log("Hello Good Mornig after 2 second"),2000);

**Output :**

****

**Q. 12 To clear SetTimeout,Setinterval**

**settimeid.js**

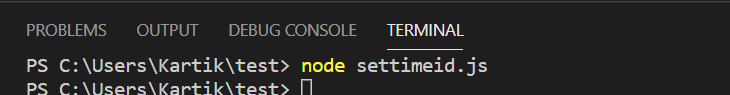
const myfunc=()=>

console.log("you cannot see this message");

const timerId=setTimeout(myfunc,0);

clearTimeout(timerId);

**Output :**

****

**Q. 13 write a program to find Prime numbers from 1 to 50**

**primenumber.js**

const prompt = require('prompt-sync')();

function prime()

{

var i, number, count;

var n = parseInt(prompt("Enter Number: "));

for(number = 1; number <= n; number++)

{

count = 0;

for (i = 2; i <= number/2; i++)

{

if(number % i == 0)

{

count++;

break;

}

}

if(count == 0 && number != 1 )

{

console.log(`${number}`);

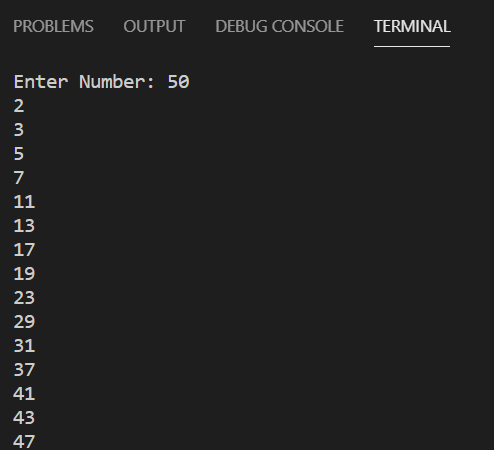
}

}

}

var r=prime();

**Output :**

****

**Q. 14 write a program to find an Amstrong number**

**amstrong.js**

const prompt=require('prompt-sync')();

function amstrong()

{

var number,check, rem, sum = 0;

number=parseInt(prompt('Enter Number: '));

check = number;

while(check != 0)

{

rem = check % 10;

sum = sum + (rem \* rem \* rem);

check = parseInt(check / 10);

}

if(sum == number)

console.log("Given number is armstrong");

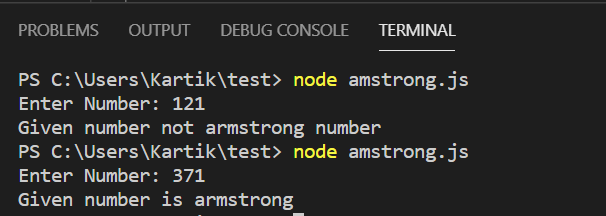
else

console.log("Given number not armstrong number");

};

var r=amstrong();

**Output :**

****

**Q.15 write a program to find factorial of a number**

**factorial.js**

const prompt=require('prompt-sync')();

function factorial()

{

var i,n,fact=1;

n=parseInt(prompt("Enter Number: "));

for(i=1;i<=n;i++)

{

fact=fact\*i;

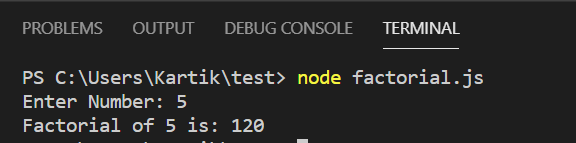
}

console.log(`Factorial of ${n} is: `+fact);

};

var r=factorial();

**Output :**

****

**PRACTICAL NO. 4**

**USING FILE HANDLING DEMONSTRATE ALL BASIC FILE OPERATIONS (CREATE, WRITE, READ, DELETE)**

**PRACTICAL NO. 4**

**USING FILE HANDLING DEMONSTRATE ALL BASIC FILE OPERATIONS (CREATE, WRITE, READ, DELETE)**

**Q. 1 Create a file to see the path of your file**

**Program:**

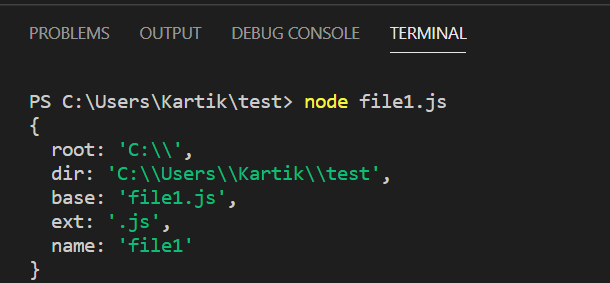
**file1.js**

const path=require("path");

const pathobj=path.parse(\_\_filename);

console.log(pathobj);

**Output :**

****

**Q.2 To create and open empty/blank file using FS FS is amodule where you can have all file operations.**

**file2.js**

const fs=require('fs');

fs.open('abc.txt','w',function(err,file)

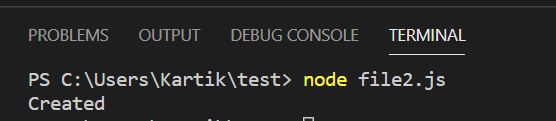
{

if(err) throw err;

console.log('Created');

});

**Output :**

****

**Q. 3 Write a program to execute the following**

**1. Write in the file**

**file3.js**

const fs=require('fs');

fs.writeFile('abc.txt','Hello, How are you',function(err,file)

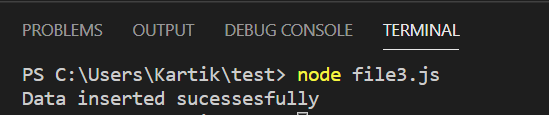
{

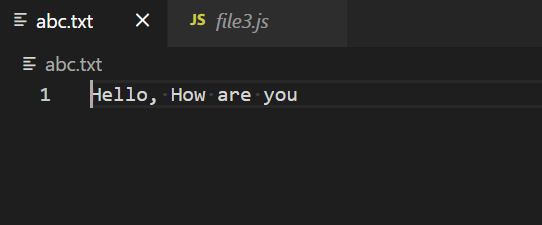
if(err) throw err;

console.log('Data inserted sucessesfully');

});

**Output :**

****

****

**2. Read a file which we have already created(if you don’t know the type of the file then it will return buffer data)**

**file4.js**

const fs=require('fs');

fs.readFile('abc.txt','utf8',function(err,data)

{

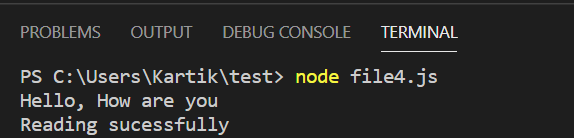
if(err) throw err;

console.log(data);

console.log('Reading sucessfully');

});

**Output :**

****

**3. Convert buffer data into original one**

**file5.js**

const fs=require('fs');

fs.readFile('abc.txt',function(err,file)

{

if(err) throw err;

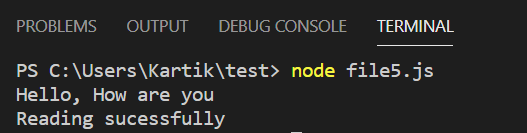
orgData=file.toString();

console.log(orgData);

console.log('Reading sucessfully');

});

**Output :**

****

**4. Append data into existing file.**

**file6.js**

const fs=require('fs');

fs.appendFile('abc.txt','\nWelcome....',function(err,file)

{

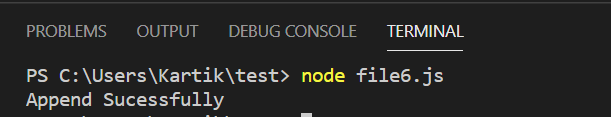
if(err) throw err;

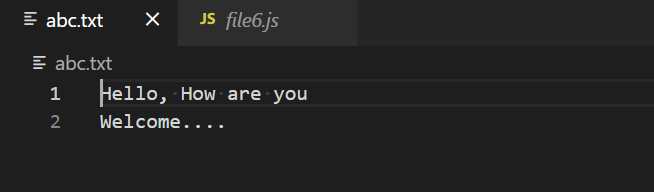
console.log('Append Sucessfully');

});

//you can use appendsync remove the call back function

**Output :**

****

****

**5. Asynchronous file appending**

**file6.js**

const fs=require('fs');

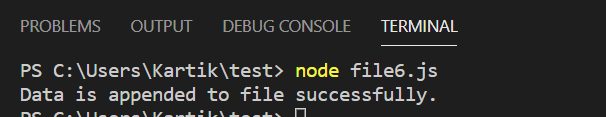
var data = "\nLearn Node.js with the help of well built Node.js Tutorial.";

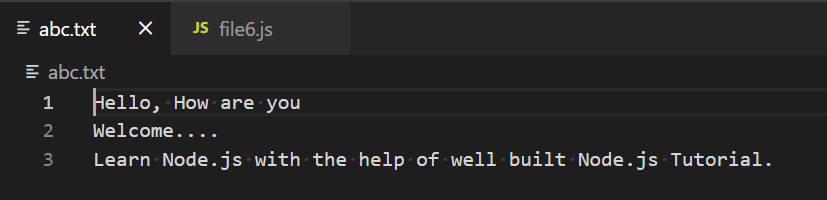
// append data to file

fs.appendFileSync('abc.txt',data, 'utf8');

console.log("Data is appended to file successfully.");

**Output :**

****

****

**6. Rename the file**

**file8.js**

const fs=require('fs');

fs.rename('abc.txt','pqr.txt',function(err,file)

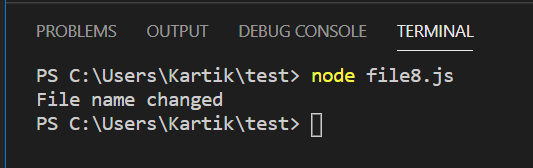
{

if(err) throw err;

console.log('File name changed');

});

**Output :**

****

**7. Delete the file**

**file7.js**

const fs=require('fs');

fs.unlink('pqr.txt',function(err,file)

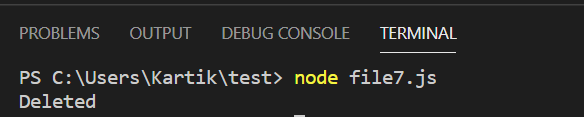
{

if(err) throw err;

console.log('Deleted');

});

**Output :**

****

**Q.4 write a program to execute the following operations**

**1. create folder with your name**

**CFolder.js**

//Creating Folder

const fs=require('fs');

fs.mkdir('kartik',function(err,file)

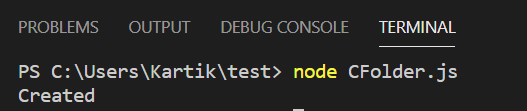
{

if(err) throw err;

console.log('Created');

});

**Output :**

****

**2. Create a file in that folder and name it as myfile.txt and add data into it.**

**CFolder.js**

//Creating myfile.txt

const fs=require('fs');

fs.open('kartik/myfile.txt','w',function(err,file)

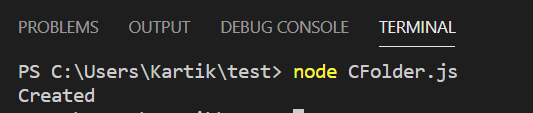
{

if(err) throw err;

console.log('Created');

});

**Output :**



**Add data into myfile.txt**

//Write in myfile.txt

const fs=require('fs');

fs.writeFile('kartik/myfile.txt','Hello...',function(err,file)

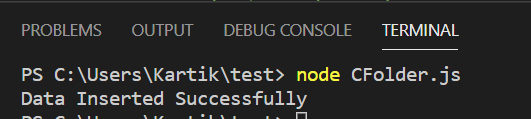
{

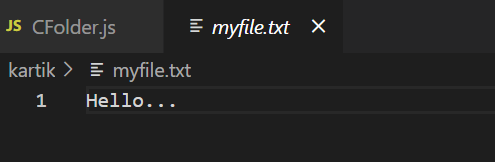
if(err) throw err;

console.log('Data Inserted Successfully');

});

**Output :**

****

****

**3. Add more data into the file.**

**CFolder.js**

//Add More data in the file

const fs=require('fs');

fs.appendFile('kartik/myfile.txt','\nWelcome.....\nHow are you..',function(err,file)

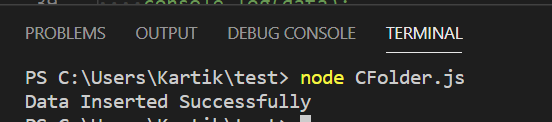
{

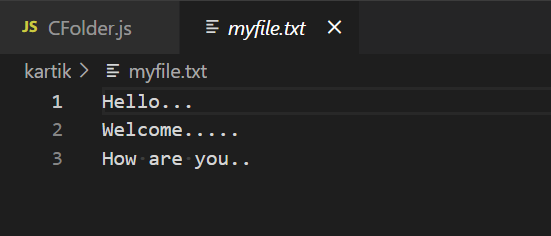
if(err) throw err;

console.log('Data Inserted Successfully');

});

Output :





**4. Read the data from the file.**

**Cfolder.js**

//read the data

const fs=require('fs');

fs.readFile('kartik/myfile.txt',function(err,file)

{

if(err) throw err;

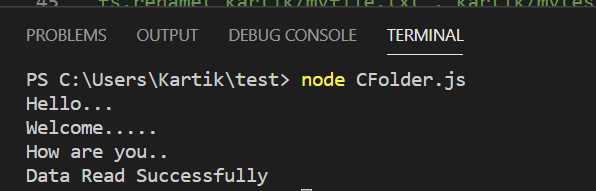
data=file.toString();

console.log(data);

console.log('Data Read Successfully');

});

**Output :**

****

**5. Rename the file as mytestfile.txt**

**CFolder.js**

//Rename file name

const fs=require('fs');

fs.rename('kartik/myfile.txt','kartik/mytestfile.txt',function(err,file)

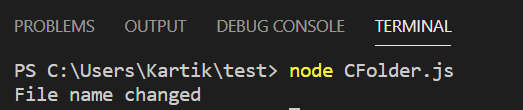
{

if(err) throw err;

console.log('File name changed');

});

**Output :**

****

**6. Delete the file and the folder**

**CFolder.js**

//delete file

const fs=require('fs');

fs.unlink('kartik/mytestfile.txt',function(err,file)

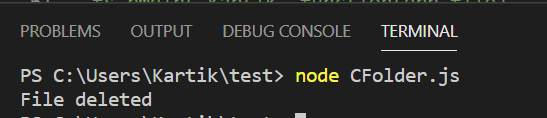
{

if(err) throw err;

console.log('File deleted');

});

**Output :**

****

**Delete the folder**

//Delete Folder

const fs=require('fs');

fs.rmdir('kartik',function(err,file)

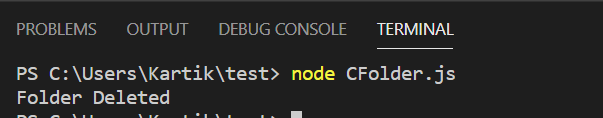
{

if(err) throw err;

console.log('Folder Deleted');

});

**Output :**

****

**PRACTICAL NO. 5**

**CREATE AN HTTP SERVER AND PERFORM OPERATIONS ON IT**

**PRACTICAL NO. 5**

**CREATE AN HTTP SERVER AND PERFORM OPERATIONS ON IT**

**Q.1 Create Http Server**

**createhttpserver.js**

var http=require('http');

var server=http.createServer(function(req,res)

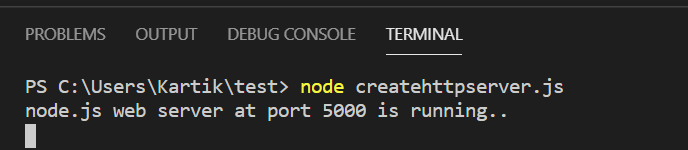
{

});

server.listen(5000);

console.log('node.js web server at port 5000 is running..');

**Output :**

****

**httpserver2.js**

var http=require('http');

var server=http.createServer(function(req,res)

{

if(req.url=='/')

{

res.writeHead(200,{'Content-Type':'text/html'});

res.write('<html><body><p>This Page.</p></body></html>');

res.end();

}

else(req.url=="/student")

{

res.writeHead(200,{'Content-Type':'text/html'});

res.write('<html><body><p>hello how are you</p></body></html>');

res.end();

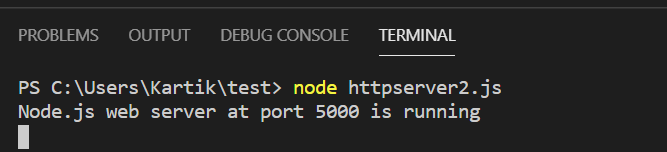
}

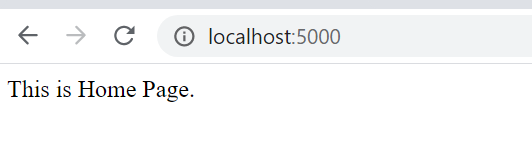
});

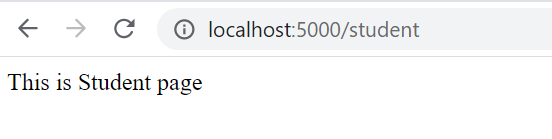
server.listen(5000);

console.log('Node.js web server at port 5000 is running');

**Output :**







**httpserver3.js**

var http = require('http');

var server = http.createServer(function (req, res) {

if (req.url == '/home')

{

res.writeHead(200, { 'Content-Type': 'text/html' });

res.write('<html><body><p>This is home Page.</p></body></html>');

res.end();

}

else if(req.url == "/student")

{

res.writeHead(200, { 'Content-Type': 'text/html' });

res.write('<html><body><p>This is student page.</p></body></html>');

res.end();

}

else if(req.url == "/aboutus")

{

res.writeHead(200, { 'Content-Type': 'text/html' });

res.write('<html><body><p>This is aboutus page.</p></body></html>');

res.end();

}

else if(req.url == "/contact")

{

res.writeHead(200, { 'Content-Type': 'text/html' });

res.write('<html><body><p>This is contact page.</p></body></html>');

res.end();

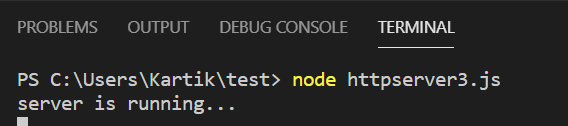
}

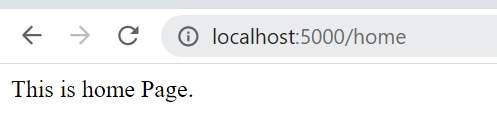
});

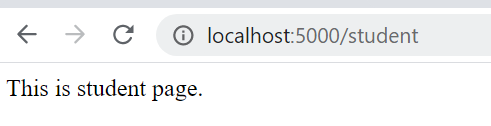
server.listen(5000);

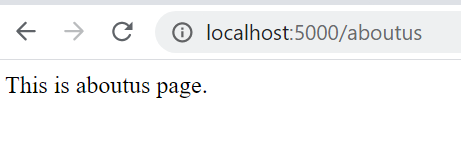
console.log('server is running...');

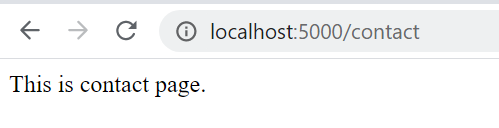
**Output :**

****

****

****

****

****

**PRACTICAL NO. 6**

**CREATE AN APPLICATION TO ESTABLISH A CONNECTION WITH THE MYSQL DATABASE AND PERFORM BASIC DATABASE OPERATIONS ON IT**

**PRACTICAL NO. 6**

**CREATE AN APPLICATION TO ESTABLISH A CONNECTION WITH THE MYSQL DATABASE AND PERFORM BASIC DATABASE OPERATIONS ON IT**

**Q.1 Installation of xammp server**

**Step 1**

To download the XAMPP server, visit the "[Apache Friends](https://www.apachefriends.org/index.html)" website in your web browser.

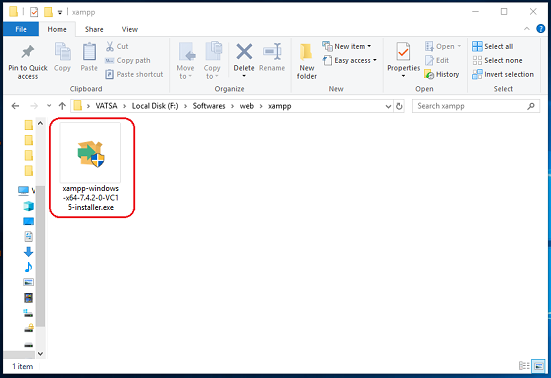
**Step 2**

Click on "XAMPP for Windows". Then, navigate the downloading location and the file will be automatically downloaded.



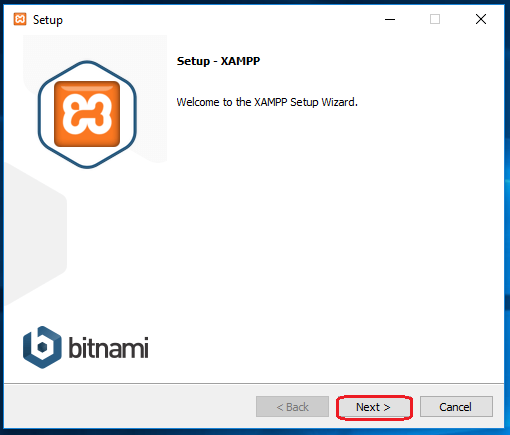
**Step 3**

Double-click the downloaded file to launch the XAMPP installer.



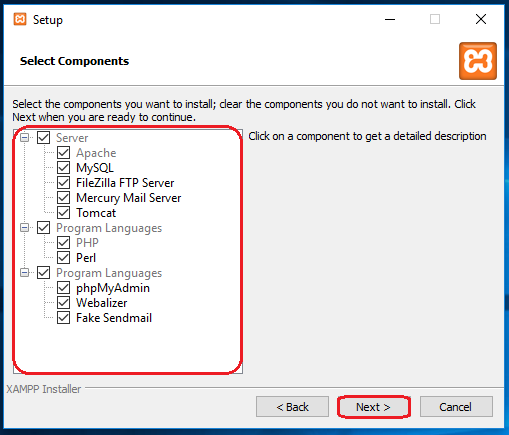
**Step 4**

"Setup" window will appear on the screen. Then, click on the "Next" button.



**Step 5**

Select the components that you want to install and click on the "Next" button.

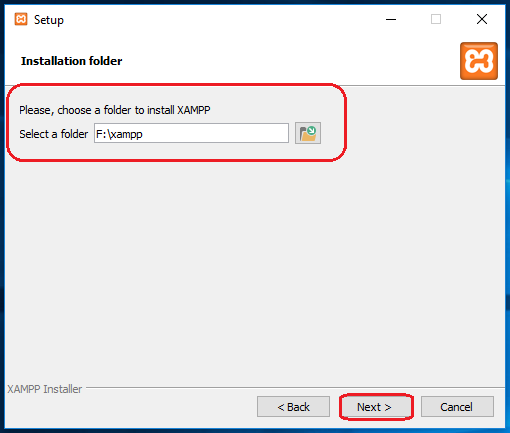


**Note**

By default, all components are selected in your XAMPP installation.

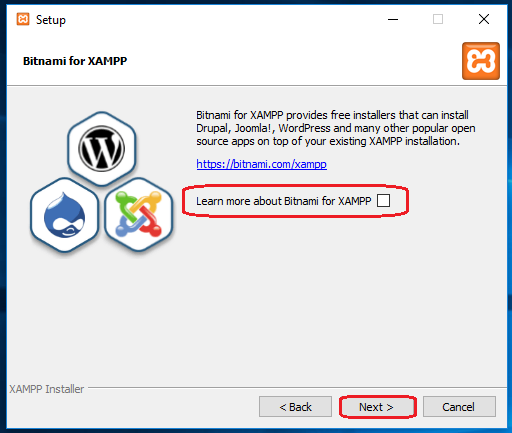
**Step 6**

Choose a folder to install the XAMPP and click on the "Next" button.



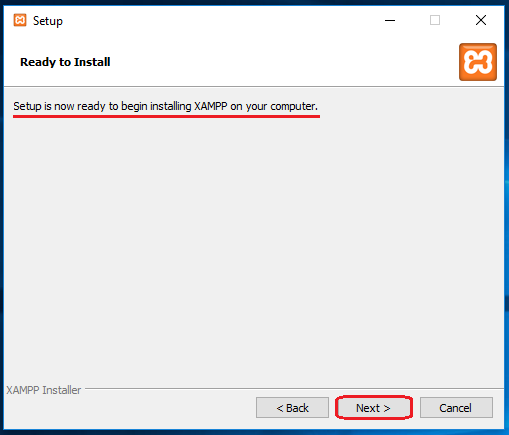
**Step 7**

Uncheck the "Learn more about Bitnami for XAMPP" option and click on the "Next" button.



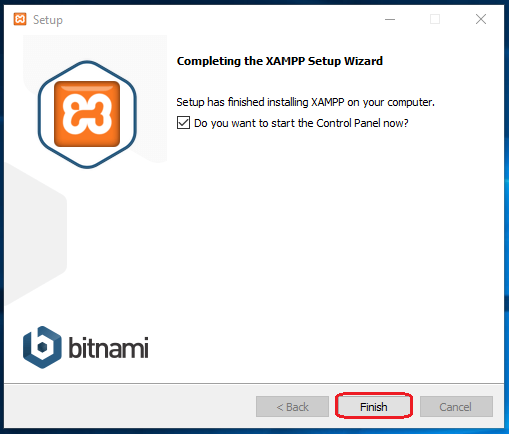
**Step 8**

"Ready to Install" window will appear on the screen, then click on the "Next" button.



**Step 9**

Click on the "Finish" button.



**Step 10**

Select a language. (either English or German) and click on the "Save" button.



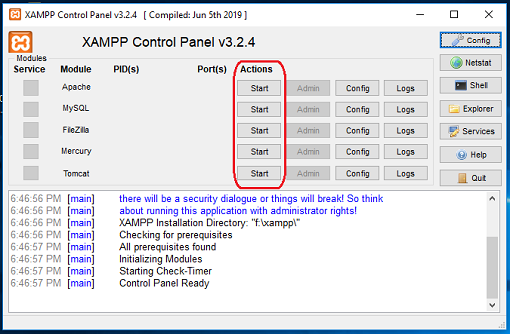
Configuration Process of XAMPP Server

**Step 1**

Start the XAMPP control panel through the "Run as administrator" option.

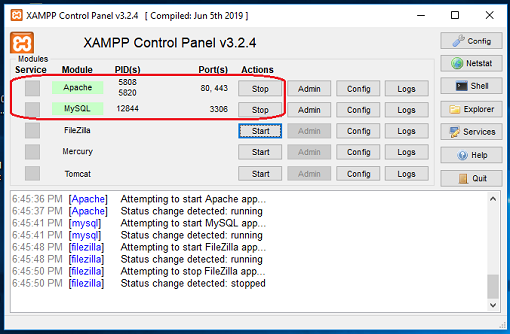
**Step 2**

"XAMPP Control Panel" will appear on the screen and click on "Start" action to start the "Apache" and "MySQL" modules.



**Note:-**

The default XAMPP server settings should work for most users. When you start the related modules (services) then, the color of the related modules (service) becomes change into the green color and the PID(s) and the Port(s) number will also be shown to the user.



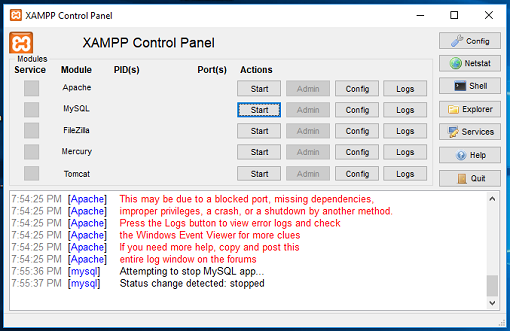
Fix an Issue of Apache not Starting in XAMPP

Some of the users may face an issue with Apache and MySQL module services not working in XAMPP. However, depending on the setup configuration or usage of your system, you may need to change the port number of the Apache and the MySQL. For example, because the "World Wide Web Services" under "Internet Information Services (IIS)" is also run on port 80 in your system, which is also the default port of apache in XAMPP. And,  the two servers (applications) can not use (run) the same port simultaneously. Follow the instructions below to fix the problem with Apache and MySQL services.

**Method 1. Change the default port of Apache**

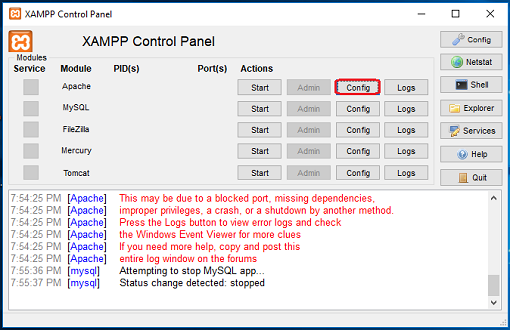
**Step 1**

Open the XAMPP Control Panel.



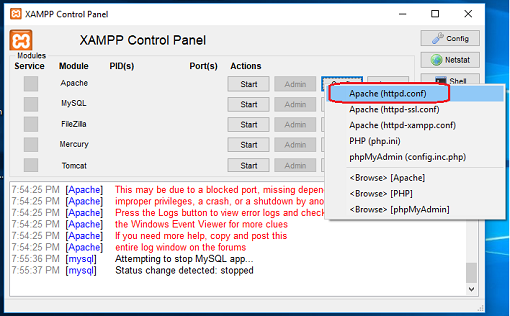
**Step 2**

In Apache Module Service, click on the "Config" button.



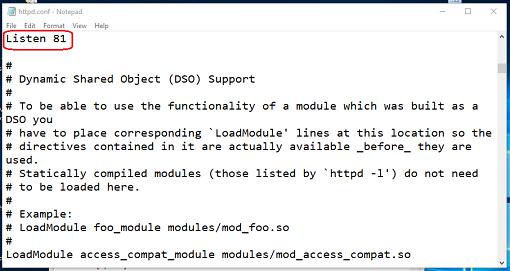
**Step 3**

Click on "Apache (httpd.conf)" option.



**Step 4**

By pressing the "Ctrl + F" key, find the "Listen 80" and replace it with another open port (like 81 or 9080) and save the file.

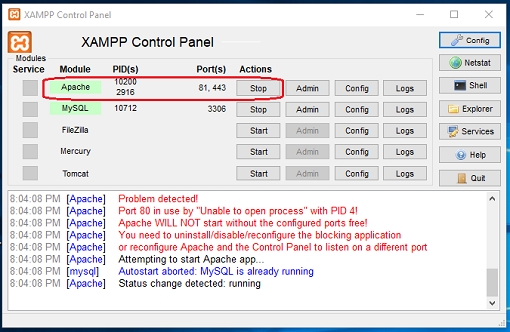


**Step 5**

Restart the XAMPP Server.

**Step 6**

Again, open the "XAMPP Control Panel" and click on the start option under the "Apache" module services.



**1. Connection**

**connectdb.js**

var mysql=require('mysql');

var con=mysql.createConnection({

host:"localhost",

user:"root",

password:""

});

con.connect(function(err)

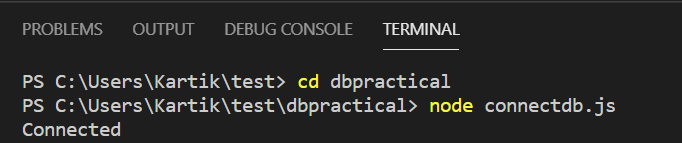
{

if(err) throw err;

console.log("Connected");

});

**Output :**

****

**2.Create datanase**

**createdb.js**

var mysql=require('mysql');

var con=mysql.createConnection({

host:"localhost",

user:"root",

password:""

});

con.connect(function(err)

{

if(err) throw err;

con.query("create database mydatabase",function(err,result)

{

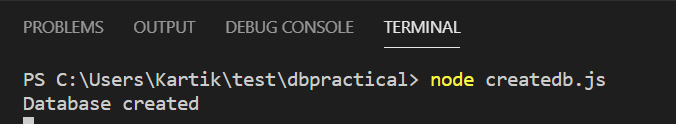
if(err) throw err;

console.log("Database created");

});

});

**Output :**

****

**3. Create table in database**

**createtable.js**

var mysql=require('mysql');

var con=mysql.createConnection({

host:"localhost",

user:"root",

password:"",

database:"mydatabase"

});

con.connect(function(err)

{

if(err) throw err;

console.log("Connected");

var sql="create table employee(e\_id int,e\_name varchar(10),e\_address varchar(10),e\_desig varchar(10))";

con.query(sql,function(err,result)

{

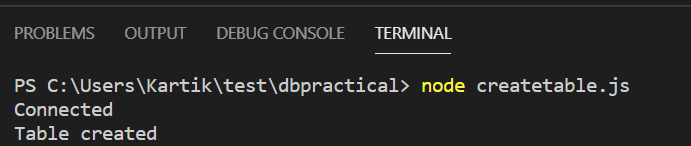
if(err) throw err;

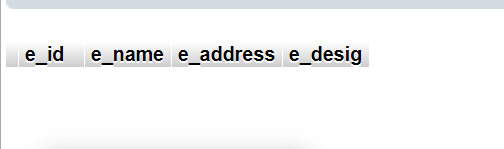
console.log("Table created");

});

});

**Output :**

****

****

**4. Insert Record in table**

**insertdb.js**

var mysql=require('mysql');

var con=mysql.createConnection

({

host:"localhost",

user:"root",

password:"",

database:"mydatabase"

});

con.connect(function(err)

{

if(err) throw err;

console.log("Connected");

var sql="insert into employee values(101,'rahul','thane','Developer'),(102,'sager','panvel','Programmer'),(103,'yash','nerul','Tester'),(104,'aarush','kalyan','Planning')";

con.query(sql,function(err,result)

{

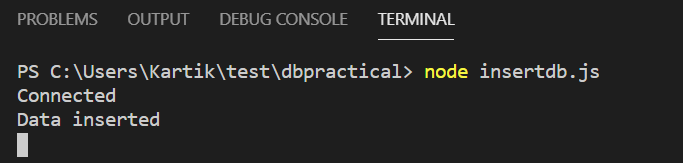
if(err) throw err;

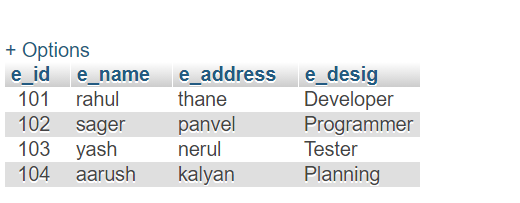
console.log("Data inserted");

});

});

**Output :**

****

****

**5. Display all records**

**display.js**

var mysql=require('mysql');

var con=mysql.createConnection({

host:"localhost",

user:"root",

password:"",

database:"mydatabase"

});

con.connect(function(err)

{

if(err) throw err;

console.log("Connected");

var sql="select \* from employee";

con.query(sql,function(err,result)

{

if(err) throw err;

console.log(result);

});

});

**Output :**

****

**6. Update Records**

**update.js**

var mysql=require('mysql');

var con=mysql.createConnection({

host:"localhost",

user:"root",

password:"",

database:"mydatabase"

});

con.connect(function(err)

{

if(err) throw err;

console.log("Connected");

var sql="update employee set e\_name='Harsh' where e\_id=102";

con.query(sql,function(err,result)

{

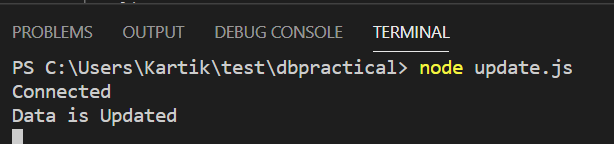
if(err) throw err;

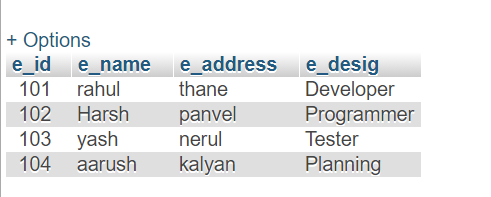
console.log("Data is Updated");

});

});

**Output :**

****

****

**7. Delete Records**

**delete.js**

var mysql=require('mysql');

var con=mysql.createConnection({

host:"localhost",

user:"root",

password:"",

database:"mydatabase"

});

con.connect(function(err)

{

if(err) throw err;

console.log("Connected");

var sql="delete from employee where e\_id=104";

con.query(sql,function(err,result)

{

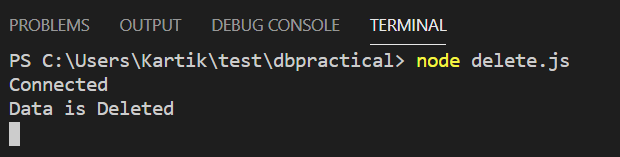
if(err) throw err;

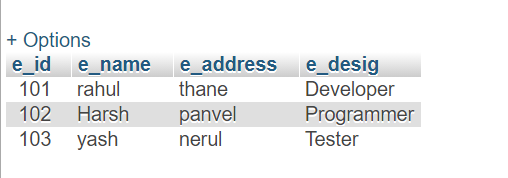
console.log("Data is Deleted");

});

});

**Output :**

****

****

**8.Select Record**

**display.js**

var mysql=require('mysql');

var con=mysql.createConnection({

host:"localhost",

user:"root",

password:"",

database:"mydatabase"

});

con.connect(function(err)

{

if(err) throw err;

console.log("Connected");

var sql="select \* from employee where e\_id=102";

con.query(sql,function(err,result)

{

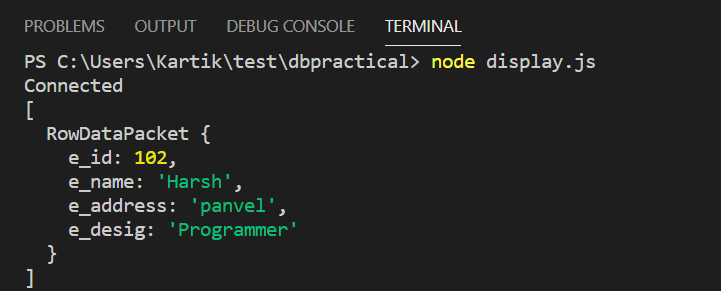
if(err) throw err;

console.log(result);

});

});

**Output :**

****

**PRACTICAL NO. 7**

**CREATE AN APPLICATION USING FILTERS**

**PRACTICAL NO. 7**

**CREATE AN APPLICATION USING FILTERS**

**PRACTICAL NO. 8**

**CREATE AN APPLICATION TO DEMONSTRATE DIRECTIVES**

**PRACTICAL NO. 8**

**CREATE AN APPLICATION TO DEMONSTRATE DIRECTIVES**

**Q.1 AngularJs directives using Array**

**Directive1.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

<script src="angular.min.js"></script>

</head>

<body ng-app="" ng-init="students=['Mayuresh','Sonal','Divya','Kartik']">

<ol>

<li ng-repeat="name in students">

{{name}}

</li>

</ol>

<div ng-repeat="name in students">

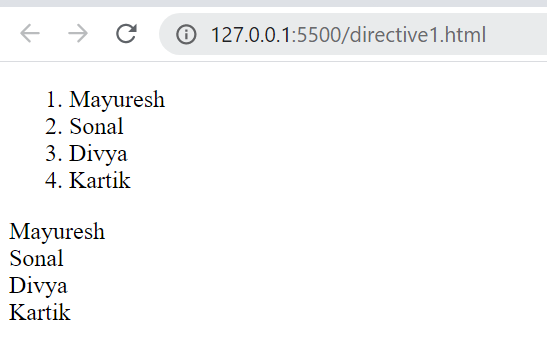
{{name}}

</div>

</body>

</html>

**Output :**

****

**Q.2 AngularJs directive using different tags**

**directive2.js**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

<script src="angular.min.js"></script>

</head>

<body ng-app ng-init="checked=true">

Click Me: <input type="checkbox" ng-model="checked"><br/>

<div>

New: <input ng-if="checked" type="text">

</div>

<div>

Read-only: <input ng-readonly="checked" type="text" value="This is read-only">

</div>

<div>

Disabled: <input ng-disabled="checked" type="text" value="this is desabled">

</div>

<div>

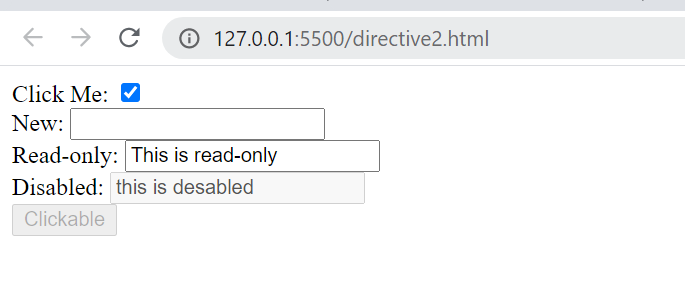
<button ng-disabled="checked">Clickable</button>

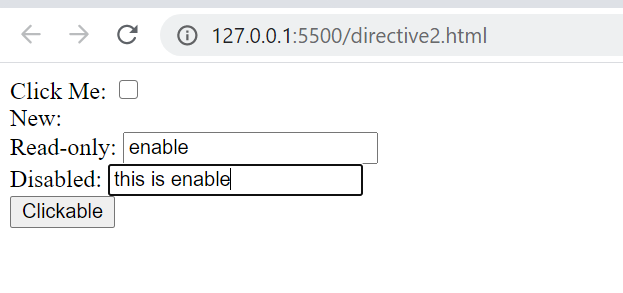
</div>

</body>

</html>

**Output :**

****

****

**PRACTICAL NO. 9**

**DEMONSTRATE CONTROLLERS IN ANGULAR.JS THROUGH AN APPLICATION**

**PRACTICAL NO. 9**

**DEMONSTRATE CONTROLLERS IN ANGULAR.JS THROUGH AN APPLICATION**

**Q.1 Controller**

**module.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

<script src="angular.min.js">

</script>

</head>

<body ng-app="myNgApp">

<div ng-controller="mycontroller">

{{message}}

{{price}}

</div>

</body>

<script>

var ngApp=angular.module("myNgApp",[]);

ngApp.controller('mycontroller',function($scope)

{

$scope.message="Hello world";

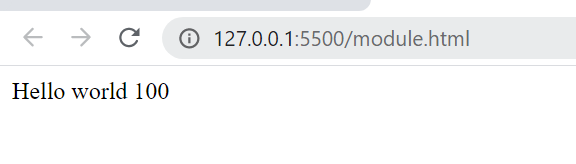
$scope.price="100";

});

</script>

</html>

**Output :**

****

**module1.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

<script src="angular.min.js">

</script>

</head>

<body ng-app="myNgApp">

<div ng-controller="mycontroller">

Enter Message: <input type="text" ng-model="message"/><br />

<button ng-click="showmsg(message)">show message</button>

</div>

<script>

var ngApp=angular.module('myNgApp',[]);

ngApp.controller('mycontroller',function($scope)

{

$scope.message="hello world";

$scope.showmsg=function(msg)

{

alert(msg);

};

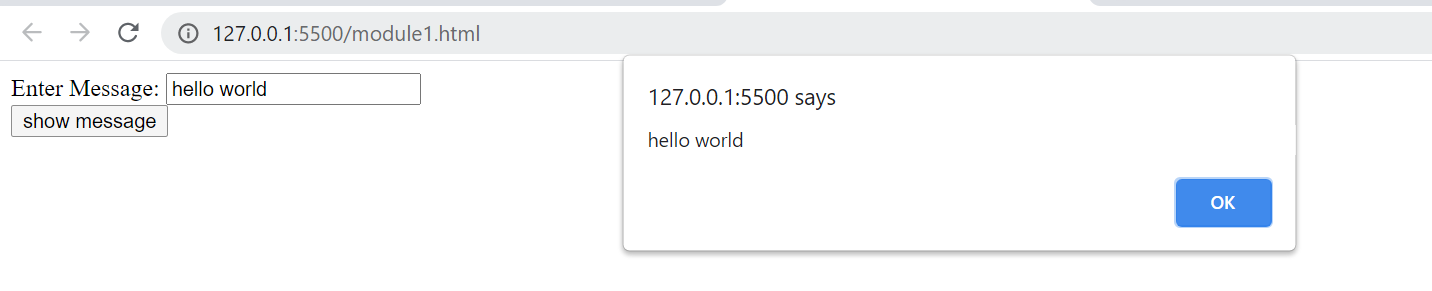
});

</script>

</body>

</html>

**Output :**

****

**Q.2 Separate js file call in html file**

**source.js**

var myApp = angular.module("MyModule",[]);

/\*myApp.controller("MyController",function($scope) {

var employee={

firstName:"Kartik",

lastName:"Chopade",

Gender:"Male"

};

$scope.employee=employee;

});

\*/

var myController=function($scope)

{

var employee={

firstName:"Kartik",

lastName:"Chopade",

Gender:"Male"

};

$scope.employee=employee;

};

myApp.controller("MyController",myController);

**controller.html**

<!DOCTYPE html>

<html lang="en" ng-app="MyModule">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

<script src="angular.min.js"></script>

<script src="source.js"></script>

</head>

<body>

<div ng-controller="MyController">

<div>

FirstName:{{employee.firstName}}

</div>

<div>

LastName:{{employee.lastName}}

</div>

<div>

Gender:{{employee.Gender}}

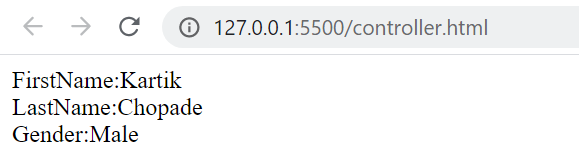
</div>

</div>

</body>

</html>

**Output :**

****

**PRACTICAL NO. 10**

**DEMONSTRATE FEATURES OF ANGULAR.JS FORMS WITH A PROGRAM**

**PRACTICAL NO. 10**

**DEMONSTRATE FEATURES OF ANGULAR.JS FORMS WITH A PROGRAM**

**Q.1 Basic Feacutes in AngularJs**

**index.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js">

</script>

</head>

<body>

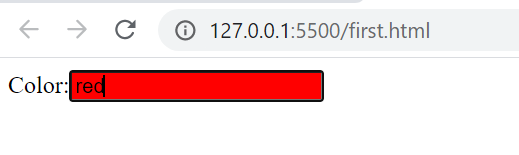
<div ng-app="" ng-init="color='green'">

Color:<input style="background-color: {{color}}" ng-model="color">

</div>

</body>

**Output :**

****

**Q.2 write a program in angular js for basic expression**

**index.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js">

</script>

</head>

<body ng-app>

<div ng-app ng-init="color='green'">

Expression:{{10/5}}

</div>

<div>

test:{{60-30}}

</div>

<div>

True or false:{{2==3}}

</div>

<div>

Subject:{{"angularjs"+" "+"javascript"}}

</div>

<div>

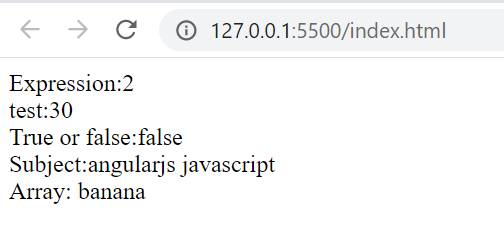
Array: {{['Apple','orange','banana'][2]}}

</div>

</body>

</html>>

**Output :**

****

**Q.3 Calculate Area**

**widthlength.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.8.2/angular.min.js">

</script>

</head>

<body bgcolor="pink">

<div ng-app="" ng-init="width=7.5;height=10.9;">

<center> Area:{{width\*height}}</center><br>

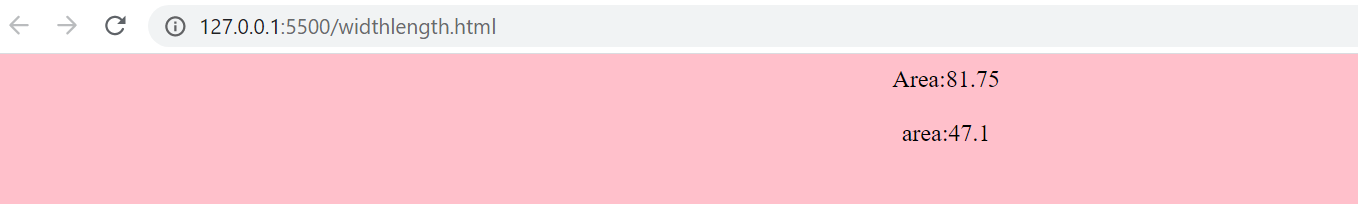
<center>area:{{3.14\*2\*width}}</center>

</div>

</body>

</html>

**Output :**

****

**PRACTICAL NO. 11**

**CREATE A SPA (SINGLE PAGE APPLICATION)**

**PRACTICAL NO. 11**

**CREATE A SPA (SINGLE PAGE APPLICATION)**