Sipna College of Engineering & Technology, Amravati.
Department Of Computer Science & Engineering

LIST OF PRACTICALS
SESSION 2021-22

Branch :- Computer Sci. & Engg. **Class :- III Year (A/B/C) Subject :- C-Skill Lab - III (5KS09)** **Semester :- Vth**

SR.No.	Aim of Practical
1.	Introduction to the Node.js and its installation to print Hello World
2.	To study built-in modules and implement the user defined built-in modules in the Node.js
3.	To study HTTP module and implement Node.js as a web server
4.	To study and implement Node.js File system module to read, write, create, update, delete and rename the file
5.	To study the URL module of the Node.js and write a program that opens the requested file and returns the content of the file to the client. If anything goes wrong, throw a 404 error.
6.	To convert the output "Hello World!" into upper-case letters by installing the “upper-case” package of NPM.
7.	To study event handling in Node.js and demonstrate it using event module and EventEmitter object.
8.	To install MySQL and its driver and create connection with it using Node.js.
9.	To demonstrate the creation database and table in MySQL using Node.js
10.	To demonstrate the insertion of single and multiple records in the MySQL using “INSERT” statement and Node.js
	New Additions
11.	To demonstrate the display of records from the MySQL database using “SELECT” statement and display it using Node.js
12.	To demonstrate the display the records based on condition from the MySQL database using “WHERE” statement using Node.js

Practical In-charge
Dr. A.V. Zade
Prof. S. R. Gudadhe
Prof. A.R. Ladole
Prof. N.G.Rathi

H.O.D.
Dr. V.K. Shandilya

Practical No: - 1

Aim: Introduction to the Node.js and its installation to print Hello World

Software Used: Node.Js
Theory:-

What is Node.js?

- Node.js is an open source server environment
- Node.js is free
- Node.js runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- Node.js uses JavaScript on the server

Why Node.js?

Node.js uses asynchronous programming!
A common task for a web server can be to open a file on the server and return the content to the client.

How Node.js handles a file request:
1. Sends the task to the computer's file system.
2. Ready to handle the next request.
3. When the file system has opened and read the file, the server returns the content to the client.

Node.js eliminates the waiting, and simply continues with the next request.
Node.js runs single-threaded, non-blocking, asynchronously programming, which is very memory efficient.

What Can Node.js Do?

- Node.js can generate dynamic page content
- Node.js can create, open, read, write, delete, and close files on the server
- Node.js can collect form data
- Node.js can add, delete, modify data in your database

What is a Node.js File?

- Node.js files contain tasks that will be executed on certain events
- A typical event is someone trying to access a port on the server
- Node.js files must be initiated on the server before having any effect
- Node.js files have extension ".js"

Download Node.js

The official Node.js website has installation instructions for Node.js: https://nodejs.org

Installation Steps

1. Download the Windows installer from Nodejs.org.
2. Run the installer (the . msi file you downloaded in the previous step.)
3. Follow the prompts in the installer (Accept the license agreement, click the NEXT button a bunch of times and accept the default installation settings).
4. Restart your computer.

Once you have downloaded and installed Node.js on your computer, let's try to display "Hello World" in a web browser.
Create a Node.js file named "Exp_1.js", and add the following code:

Program:

```
console.log('Hello World')
```

Output:

Hello World!

Conclusion:

Viva Questions:

1. What are the features of Node.js?
2. What is Node Js ?
3. when to use Node.js?
4. How node.js works?
5. What are the main advantages of using Node.js?

Practical No:- 2

Aim: To study built-in modules and implement the user defined built-in modules in the Node.js

Software Used: Node.js
Theory:

What is a Module in Node.js?
Modules to be the same as JavaScript libraries. A set of functions you want to include in your application.

Built-in Modules

Node.js has a set of built-in modules which you can use without any further installation.

Module	Description
assert	Provides a set of assertion tests
buffer	To handle binary data
child_process	To run a child process
cluster	To split a single Node process into multiple processes
crypto	To handle OpenSSL cryptographic functions
dgram	Provides implementation of UDP datagram sockets
dns	To do DNS lookups and name resolution functions
domain	Deprecated. To handle unhandled errors
events	To handle events
fs	To handle the file system
http	To make Node.js act as an HTTP server
https	To make Node.js act as an HTTPS server.
net	To create servers and clients
os	Provides information about the operation system
path	To handle file paths
punycode	Deprecated. A character encoding scheme
querystring	To handle URL query strings
readline	To handle readable streams one line at the time
stream	To handle streaming data
string_decoder	To decode buffer objects into strings

timers	To execute a function after a given number of milliseconds
tls	To implement TLS and SSL protocols
Tty	Provides classes used by a text terminal
url	To parse URL strings
util	To access utility functions
v8	To access information about V8 (the JavaScript engine)
vm	To compile JavaScript code in a virtual machine
zlib	To compress or decompress files

Include Modules

To include a module, use the require() function with the name of the module:

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

Now your application has access to the HTTP module, and is able to create a server:

```
http.createServer(function (req, res)
{
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.end('Hello World!');
}).listen(8080);
```

User-Defined Modules

You can create your own modules, and easily include them in your applications. The following example creates a module that returns a date and time object:

Example

Create a module that returns the current date and time:

```
exports.myDateTime = function ()
{
  return Date();
};
```

Use the exports keyword to make properties and methods available outside the module file.
Save the code above in a file called "myfirstmodule.js"

Program:

Output:

Conclusion:

Viva Question:

1. **What are the modules in node JS?**
2. **What is NPM?**
3. **Why is Node.js preferred over other backend technologies like Java and PHP?**
4. **What are some of the most commonly used libraries in Node.js**

Practical No:- 3

Aim: To study HTTP module and implement Node.js as a web server.

Software Used: Node.js

Theory:
The Node.js framework is mostly used to create server-based applications. The framework can easily be used to create web servers which can serve content to users.
There are a variety of modules such as the “http” and “request” module, which helps in processing server related requests in the web server space. We will have a look at how we can create a basic web server application using Node js.

To access web pages of any web application, you need a web server. The web server will handle all the http requests for the web application e.g IIS is a web server for ASP.NET web applications and Apache is a web server for PHP or Java web applications.
Node.js provides capabilities to create your own web server which will handle HTTP requests asynchronously. You can use IIS or Apache to run Node.js web application but it is recommended to use Node.js web server.

Program:

```
var http=require('http');
http.createServer(function(req,res)
{
    res.write("Hello World");
    res.end();
}
).listen(8080);
```

Output:

Conclusion:

Viva Question:
1. What is the command used to import external libraries?
2. Explain the purpose of module.exports?

Practical No:- 4

Aim: To study and implement Node.js File system module to read, write, create, update, delete and rename the file.

Software Used: Node.js

Theory:

The Node.js file system module allows you to work with the file system on your computer.
To include the File System module, use the require() method:
var fs = require('fs');
Common use for the File System module:

- Read files
- Create files
- Update files
- Delete files
- Rename files

Read Files

The fs.readFile() method is used to read files on your computer.

Create Files

The File System module has methods for creating new files:

- fs.appendFile()
- fs.open()
- fs.writeFile()

The fs.appendFile() method appends specified content to a file. If the file does not exist, the file will be created

The fs.open() method takes a "flag" as the second argument, if the flag is "w" for "writing", the specified file is opened for writing. If the file does not exist, an empty file is created.

The fs.writeFile() method replaces the specified file and content if it exists. If the file does not exist, a new file, containing the specified content, will be created.

Update Files

The File System module has methods for updating files:

- fs.appendFile()
- fs.writeFile()

The fs.appendFile() method appends the specified content at the end of the specified file

The fs.writeFile() method replaces the specified file and content

Delete Files

To delete a file with the File System module, use the fs.unlink() method.
The fs.unlink() method deletes the specified file

Rename Files

To rename a file with the File System module, use the fs.rename() method.
The fs.rename() method renames the specified file.

Program:

```
var fs=require('fs')

fs.appendFile('test.txt', 'Contents of test file', function(err)
{
    if (err) throw err;
    console.log('File is saved successfully!');
});


var fs=require('fs')

fs.writeFile('test.txt', 'Welcome to this file from node js', function(err)
{
    if (err) throw err;
    console.log('Saved!');
});

var fs=require('fs')

fs.rename('test2.txt', 'test.txt', function(err)
{
    if (err) throw err;
    console.log('File Renamed Successfully!');
});

var fs=require('fs')

fs.unlink('test.txt', function(err)
{
    if (err) throw err;
    console.log('File Deleted Successfully!');
});
```

Output:

Conclusion:

Viva Question:

1. What is the difference between ReadFile and createReadStream functions?
2. How do you handle uncaught exceptions in node.js?

Practical No:-5

Aim: To study the URL module of the node.js and write a program that opens the requested file and returns the content of the file to client. If anything goes wrong, throw a 404 error.

Software Used: Node.js

Theory:

The URL module splits up a web address into readable parts. To include the URL module, use the require() method:

var url = require('url');

The URL module splits up a web address into readable parts.

To include the URL module, use the require() method:

var url = require('url');

Parameters: This method accepts two parameters as mentioned above and described below:

input <string>: It is the input which is string type that is used to parse the absolute or relative input URL. The base is required if the input is relative and ignored if the input is absolute.
base <string> | <URL>: It is the base URL which is either of string type or URL, used to resolve against if the input is absolute or not.
Return Value: It returns the new URL generated along with an array of data like hostname, protocol, pathname, etc.

Program:

```
var http=require('http');
var fs=require('fs');
var server=http.createServer(function(req,res){
    fs.open('input.txt','r+',function(err,fd)
    {
        if(err){
            console.error(err);
            return res.end("404 file not found");
        }

    else {
        console.log("file open successfully");


        fs.readFile('sample.txt',function(err,data)
        {
            if(!err)
```

```
        console.log('success');
        res.end(data);
        fs.close(fd);
        });
    }
});
});
server.listen(5000);
```

Output:

Conclusion:

Viva Question:

1. How would you use a URL module in Node.js?
2. What is the difference between host and hostname?

Practical No:- 6

Aim: To convert the output “Hello World!” into “upper-case” letter by installing upper case package of NPM

Software Used: Node.js

Theory:

What is NPM?

NPM is a package manager for Node.js packages, or modules if you like. The NPM program is installed on your computer when you install Node.js. NPM is already ready to run on your computer!

What is a Package?

A package in Node.js contains all the files you need for a module. Modules are JavaScript libraries you can include in your project.

Steps to download Package:

Download "upper-case":

C:\Users*Your Name*>npm install upper-case

NPM creates a folder named "node_modules", where the package will be placed. All packages you install in the future will be placed in this folder.

My project now has a folder structure like this:

C:\Users*My Name*\node_modules\upper-case

Once the package is installed, it is ready to use.

var uc = require('upper-case');

Program:

```
var http=require('http');
var uc=require('upper-case');
http.createServer(function(req,res){
res.writeHead(200,{'content-Type':'text/html'});
res.write(uc.upperCase("Hello neha"));
res.end();
}).listen(8080);
```

Ouput:

Conclusion:

Viva Question:

1. How do you manage packages in your node.js project?

2. What is the full form of NPM?

3. How to install NPM Package?

Practical No: - 7

Aim: To study event handling in Node.js and demonstrate it using event module and EventEmitter object.

Software Used: Node.js

Theory:

Events in Node.js

Every action on a computer is an event. Like when a connection is made or a file is opened. Objects in Node.js can fire events, like the readStream object fires events when opening and closing a file.

Events Module

Node.js has a built-in module, called "Events", where you can create-, fire-, and listen for- your own events. To include the built-in Events module use the require() method. In addition, all event properties and methods are an instance of an EventEmitter object. To be able to access these properties and methods, create an EventEmitter object.

The EventEmitter Object

You can assign event handlers to your own events with the EventEmitter object. To fire an event, use the emit() method.

Program:

```
const{EventEmitter}=require('events');
const emitter=new EventEmitter();

emitter.on('messageLogged',function(){
    console.log('Listner called');
});

emitter.emit('messageLogged');
```

Output:

Conclusion:

Viva Question:

3. What is EventEmitter in node.js?

4. What is EventEmitter class?

5. What does EventEmitter do?

Practical No: - 8

Aim: To install MySQL and its driver and create connection with it using Node.js.

Software Used: Node.js

Theory:

To be able to experiment with the code examples, you should have MySQL installed on your computer.
You can download a free MySQL database at https://www.mysql.com/downloads/.
Once you have MySQL up and running on your computer, you can access it by using Node.js.
To access a MySQL database with Node.js, you need a MySQL driver. This tutorial will use the "mysql" module, downloaded from NPM.
To download and install the "mysql" module, open the Command Terminal and execute the following:

npm install --save mysql

Now you have to create a mysql connection, which you can later query.
Now you have downloaded and installed a mysql database driver.
Node.js can use this module to manipulate the MySQL database:

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

Create Connection

Start by creating a connection to the database.
Use the username and password from your MySQL database.

Program:

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

var conn=mysql.createConnection({
    host:"localhost",
    user:"root",
    password:"123456"
});

conn.connect(function(err){
    if(err)
        throw err;
    console.log("connection with the mysql is successfull..");
});
```

Output:

Conclusion:

Viva Question:

1. What is MYSQL?
2. How to install MYSQL driver?
3. How to use MYSQL in Node.js?

Practical No. : - 9

Aim: To demonstrate the creation database and table in MySQL using Node.js
Software Used: Node.js
Theory:

Creating a Database

To create a database in MySQL, use the "CREATE DATABASE" statement:

Program:

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

var conn=mysql.createConnection({
    host:"localhost",
    user:"root",
    password:"123456",
    database:"mydb"
});

conn.connect(function(err){
    if(err)
        throw err;
    console.log("Database is Connected!");
    var sql= "CREATE TABLE employee (name VARCHAR(255), address VARCHAR(255))";
 //Query
    conn.query(sql,function(err,result){
    if(err)
      throw err;
    console.log("Employee Table is Created");
    });
});
```

Output:

Conclusion:

Viva Question:
1. Write down syntax for creating database?
2. Write down syntax for creating table
3. Is it possible to drop database in node.js

Practical No. : - 10

Aim: To demonstrate the insertion of single and multiple records in the MySQL using “INSERT” statement and Node.js

Software Used: Node.js

Theory:

Insert Into Table

To fill a table in MySQL, use the "INSERT INTO" statement.

Example

Insert a record in the "customers" table:

Program:

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

var conn=mysql.createConnection({
    host:"localhost",
    user:"root",
    password:"123456",
    database:"mydb"
});

conn.connect(function(err){
    if(err)
        throw err;
    console.log("Database is Connected!");
    var sql= "INSERT INTO employee (name, address) VALUES ?";

    var VALUES =[ ['Dhoni','Ranchi'],
                  ['Virat','Delhi'],
                  ['Shreyash','Mumbai'],
                  ['Rohit','Mumbai']
                ];

    conn.query(sql,[VALUES],function(err,result){
    if(err)
      throw err;
    console.log("Number of Records are inserted");
    });
});
```

Output:

Conclusion:

Viva Question:

1. Is it possible to insert multiple records into table?
2. What is record?
3. Explain steps to insert records into table.

Practical No. : - 11

Aim: To demonstrate the display of records from the MySQL database using “SELECT” statement and display it using Node.js

Software Used: Node.js

Theory:

Selecting From a Table

To select data from a table in MySQL, use the "SELECT" statement.

Example

Program:

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

var conn=mysql.createConnection({
    host:"localhost",
    user:"root",
    password:"123456",
    database:"mydb"
});

conn.connect(function(err){
    if(err)
        throw err;
    conn.query("SELECT * FROM employee",function (err, result, fields){
        if(err)
          throw err;
        console.log(result);
        });
    });
```

Output:

Conclusion:

Viva Question:

1. Which is the scripting/programming language used for Node.js application programming?
2. How is Node.js different from previous server side programming frameworks?
3. What is event-driven programming?

Practical No. : - 12

Aim: To demonstrate deletion of records from database using “DELETE” statement & updating existing records in a table by using the "UPDATE" statement and Node.js
Software Used: Node.js
Theory:
Delete Record
You can delete records from an existing table by using the "DELETE FROM" statement:
Program:

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

var conn=mysql.createConnection({
    host:"localhost",
    user:"root",
    password:"123456",
    database:"mydb"
});

conn.connect(function(err){
    if(err)
        throw err;
    conn.query("SELECT * FROM employee WHERE name='Sachin'",function (err, result, fields){
        if(err)
          throw err;
        console.log(result);
        });
    });
```

Output:

Conclusion:

Viva Question:
1.What is npm?
2.How do you connect your Node.js application to MySQL?
3.What are the features of Node.js?