**Batch: A2 Roll No.: 16010121045**

**Experiment No. 5**

|  |
| --- |
| **Title:**  React Hooks |

**Title**: Implementation of Node.js

**Problem statement:**

Consider the basic concepts, which are useful in the creation of an application.

Considering the following points, demonstrate the functionality of each with a simple script

**1) Basic Routing:**

          1. Build First server application using http module

           2. Basic routing: Demonstrate it using simple HTML/Json file

          3. Demonstrate the callback in node.js

 2)  **File operation**

          - Check Permissions of a File or Directory.

          - Checking if a file or a directory exists.

          - Determining the line count of a text file.

          - Reading a file line by line.

          - See the file content through browser.

**3) Building your custom modules**

          -To demonstrate this use some mathematics function to create custom module.

**$) Blocking and Non Blocking**

Code:

Index.js

const http = require('http');

const myMath = require('./myMath.js');

const server = http.createServer((req, res) => {

res.write(

'<p>The greatest common divisor of 5 and 15 is ' +

myMath.gcd(5, 15) +

'</p>'

);

res.write(

'<p>The least common multiple of 5 and 15 is ' + myMath.lcm(5, 15) + '</p>'

);

res.end();

});

server.listen(3000, () => {

console.log('Server is running on port 3000');

});

File Operations

const http = require('http');

const myMath = require('./myMath.js');

const fs = require('fs');

const server = http.createServer((req, res) => {

const filePath = 'myMath.js';

fs.readFile(filePath, 'utf8', (err, data) => {

*if* (err) {

res.writeHead(404, { 'Content-Type': 'text/plain' });

res.end('File not found.');

} *else* {

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

res.end(data);

}

});

});

server.listen(3000, () => {

console.log('Server is running on port 3000');

});

myMath.js

module.exports = {

gcd: function (a, b) {

*if* (b == 0) {

*return* a;

}

*return* *this*.gcd(b, a % b);

},

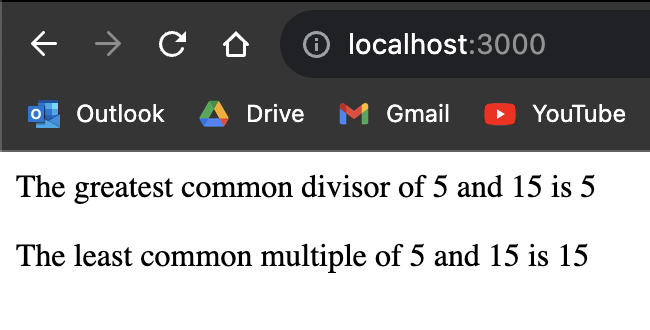
lcm: function (a, b) {

*return* (a \* b) / *this*.gcd(a, b);

},

};

Output:



File reading

