

**Batch: D-2**

**Roll No.: 16010122151**

**Experiment 2**

**Grade: AA / AB / BB / BC / CC / CD / DD**

**Signature of the Staff In-charge with date**

**Title: Node js Implementation**

**AIM:** Node JS Implementation

**Problem Definition:**

-Demonstrate the Concept of Nodejs With the help of Example.

\*(Students have to perform the task assigned within group and demonstrate the same).

**Resources used:**

---

**Expected OUTCOME of Experiment:**

**CO 1:** Build full stack applications in JavaScript using the MERN technologies.

---

**Books/ Journals/ Websites referred:**

1. Shelly Powers Learning Node O' Reilly 2 nd Edition, 2016.

**Pre Lab/ Prior Concepts:**

## **1) File operation**

- **CRUD operations**
- **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.**

## **2) Building your custom modules**

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

## **3) 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**

## **4) Blocking and Non Blocking**

## Implementation Details:

### 1) File operation

- **CRUD operations**
- **Check Permissions of a File or Directory.**
- **Checking if a file or a directory exists.**
- **Determining the line count of a text file.**

#### Code 1:

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

fs.writeFile('bruh.txt', 'Hello, world!', (err) => {
  if (err) throw err;
  console.log('File has been created and data written!');
});

fs.readFile('bruh.txt', 'utf8', (err, data) => {
  if (err) throw err;
  console.log(data);
});

fs.appendFile('bruh.txt', ' More text!', (err) => {
  if (err) throw err;
  console.log('Text has been appended to file!');
});

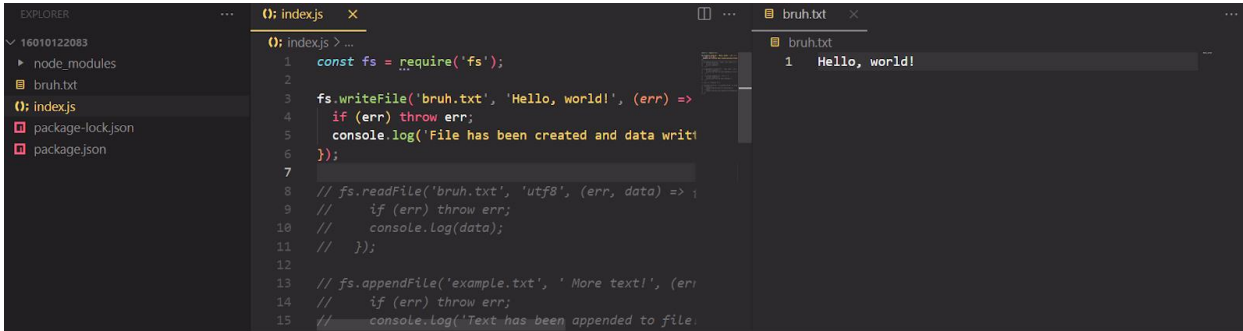
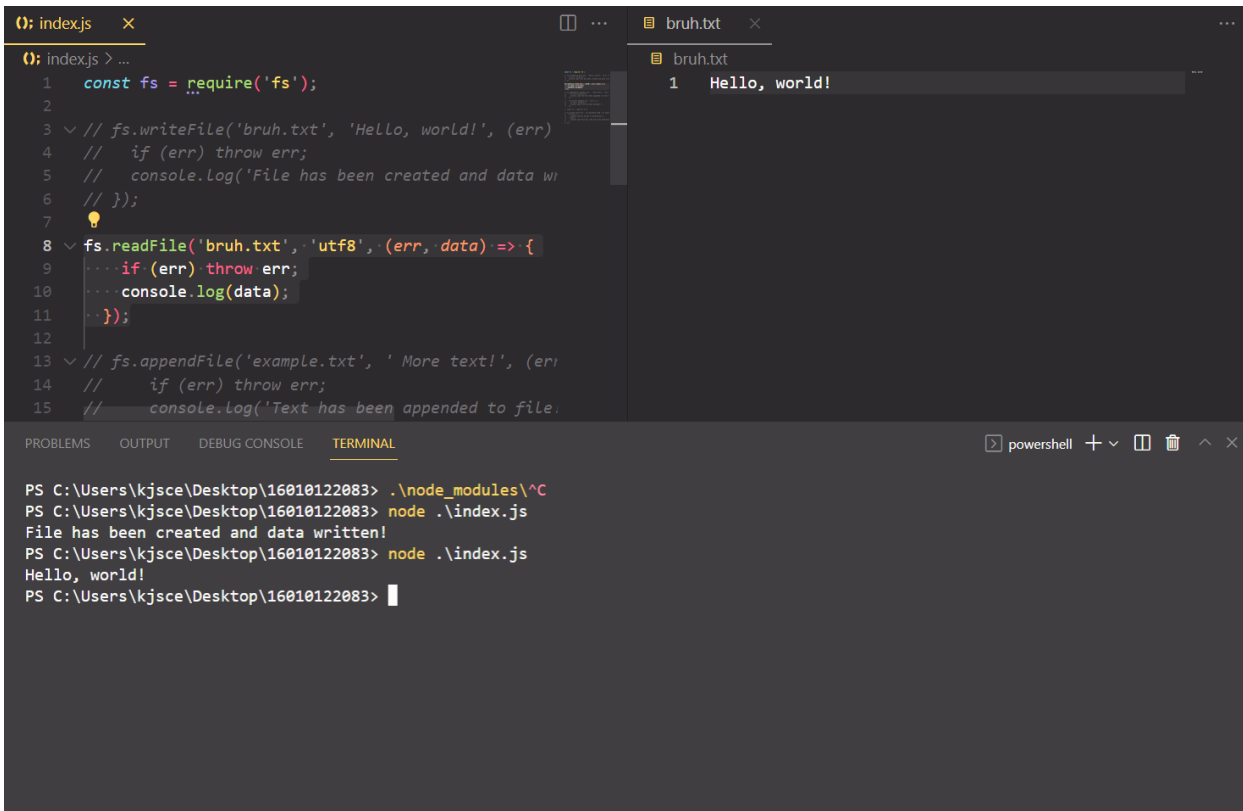
fs.unlink('bruh.txt', (err) => {
  if (err) throw err;
  console.log('File has been deleted!');
});

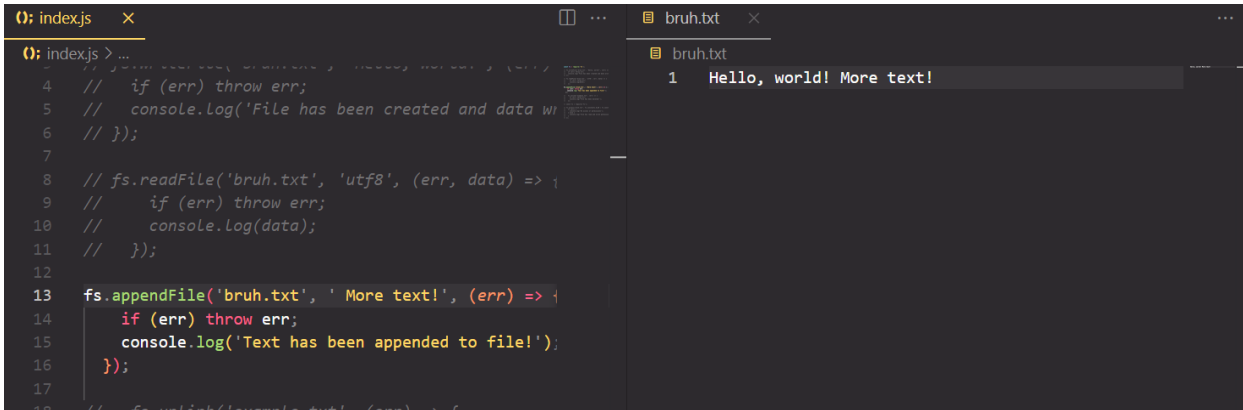
const fs = require('fs');

fs.access('mern.txt', fs.constants.R_OK | fs.constants.W_OK, (err) => {
  if (err) {
    console.log('No access or permissions');
  } else {
    console.log('File has read and write permissions');
  }
});
```

```
fs.access('example.txt', fs.constants.F_OK, (err) => {
  if (err) {
    console.log('File does not exist');
  } else {
    console.log('File exists');
  }
});
```

## Output:



The screenshot shows two files in VS Code. The left pane shows `index.js` with the following code:

```

1 // if (err) throw err;
2 // console.log('File has been created and data written!');
3 // });
4
5 // fs.readFile('bruh.txt', 'utf8', (err, data) => {
6 //   if (err) throw err;
7 //   console.log(data);
8 // });
9
10 fs.appendFile('bruh.txt', ' More text!', (err) => {
11   if (err) throw err;
12   console.log('Text has been appended to file!');
13 });
14
15 // fs.unlink('example.txt', (err) => {

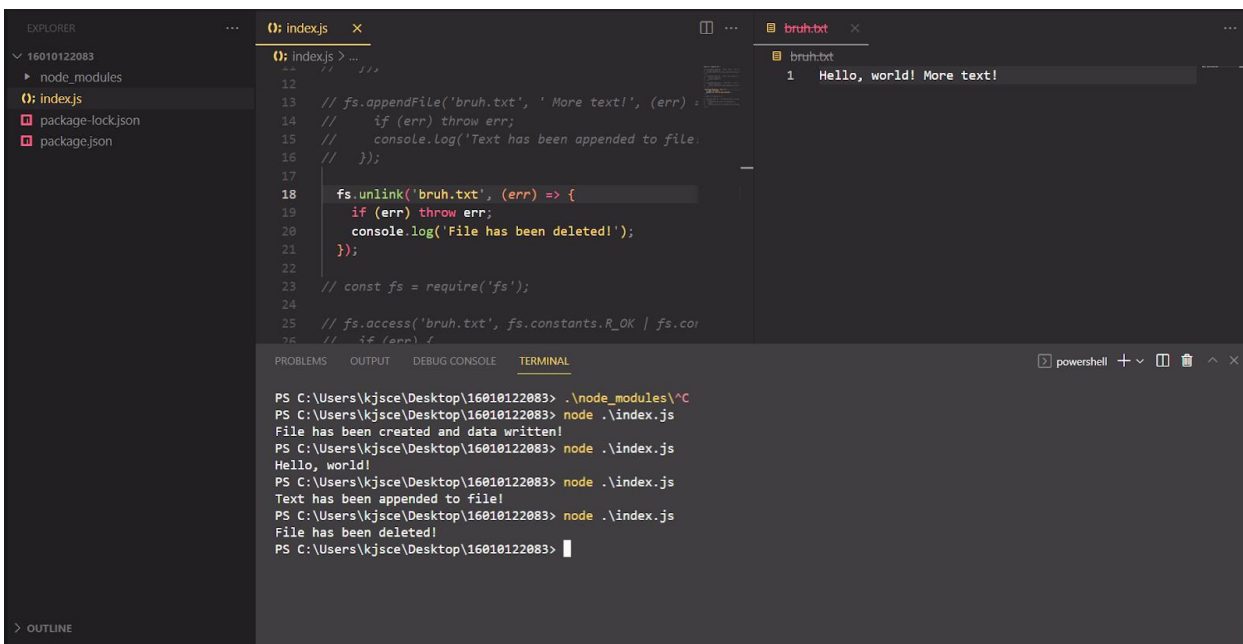
```

The right pane shows `bruh.txt` with the following content:

```

1 Hello, world! More text!

```



The screenshot shows the VS Code interface with the Explorer pane on the left showing the file structure. The main editor shows `index.js` with the following code:

```

12
13 // fs.appendFile('bruh.txt', ' More text!', (err) => {
14 //   if (err) throw err;
15 //   console.log('Text has been appended to file!');
16 // });
17
18 fs.unlink('bruh.txt', (err) => {
19   if (err) throw err;
20   console.log('File has been deleted!');
21 });
22
23 // const fs = require('fs');
24
25 // fs.access('bruh.txt', fs.constants.R_OK | fs.constants.W_OK, (err) => {
26 //   if (err) {

```

The right pane shows `bruh.txt` with the following content:

```

1 Hello, world! More text!

```

The bottom pane shows the terminal output:

```

PS C:\Users\kjsce\Desktop\16010122083> .\node_modules\node .\index.js
File has been created and data written!
PS C:\Users\kjsce\Desktop\16010122083> node .\index.js
Hello, world!
PS C:\Users\kjsce\Desktop\16010122083> node .\index.js
Text has been appended to file!
PS C:\Users\kjsce\Desktop\16010122083> node .\index.js
File has been deleted!
PS C:\Users\kjsce\Desktop\16010122083>

```

Code :Reading a file line by line.

```
fs.access('mern.txt', fs.constants.F_OK, (err) => {
  if (err) {
    console.log('File does not exist');
  } else {
    console.log('File exists');
  }
});

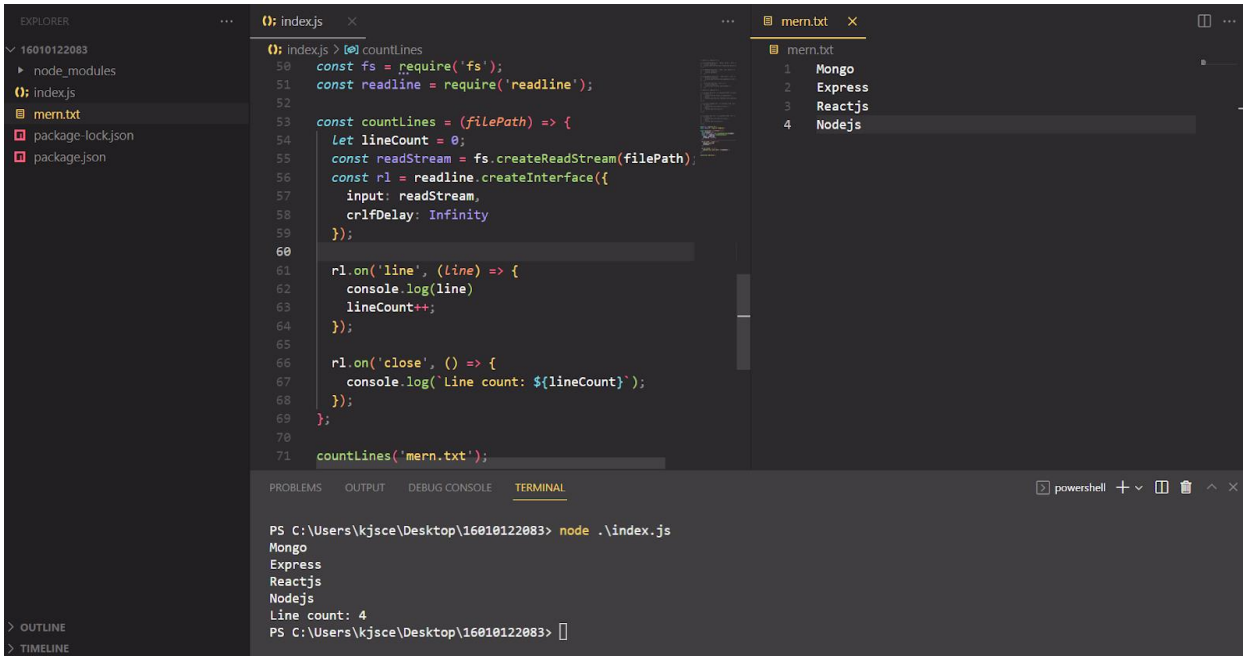
const countLines = (filePath) => {
  let lineCount = 0;
  const readStream = fs.createReadStream(filePath);
  const rl = readline.createInterface({
    input: readStream,
    crlfDelay: Infinity
  });

  rl.on('line', (line) => {
    console.log(line)
    lineCount++;
  });

  rl.on('close', () => {
    console.log(`Line count: ${lineCount}`);
  });
};

countLines('mern.txt');
```

Output 2:



```
0: index.js > countLines
50 const fs = require('fs');
51 const readline = require('readline');
52
53 const countLines = (filePath) => {
54   let lineCount = 0;
55   const readStream = fs.createReadStream(filePath);
56   const rl = readline.createInterface({
57     input: readStream,
58     crlfDelay: Infinity
59   });
60
61   rl.on('line', (line) => {
62     console.log(line);
63     lineCount++;
64   });
65
66   rl.on('close', () => {
67     console.log(`Line count: ${lineCount}`);
68   });
69 };
70
71 countLines('mern.txt');
```

```
PS C:\Users\kjsce\Desktop\16010122083> node .\index.js
Mongo
Express
Reactjs
Nodejs
Line count: 4
PS C:\Users\kjsce\Desktop\16010122083>
```



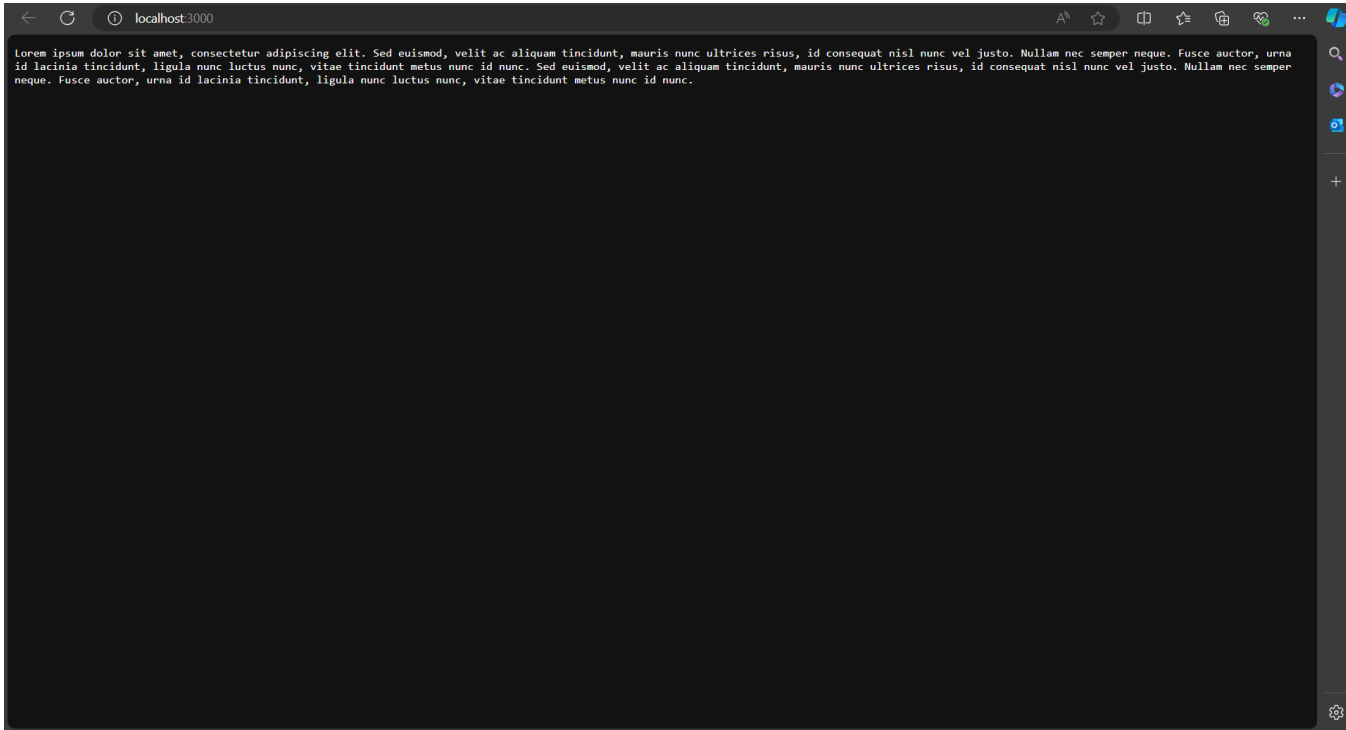
Code : See the file content through browser.

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

const PORT = 3000;
const FILE_PATH = path.join(__dirname, 'file.txt');
const server = http.createServer((req, res) => {
  if (req.url === '/' && req.method === 'GET') {
    fs.readFile(FILE_PATH, 'utf8', (err, data) => {
      if (err) {
        res.writeHead(500, {'Content-Type': 'text/plain'});
        res.end('Internal Server Error');
        return;
      }
      res.writeHead(200, {'Content-Type': 'text/plain'});
      res.end(data);
    });
  } else {
    res.writeHead(404, {'Content-Type': 'text/plain'});
    res.end('Not Found');
  }
});
server.listen(PORT, () => {
  console.log(`Server is listening on http://localhost:${PORT}`);
});
```

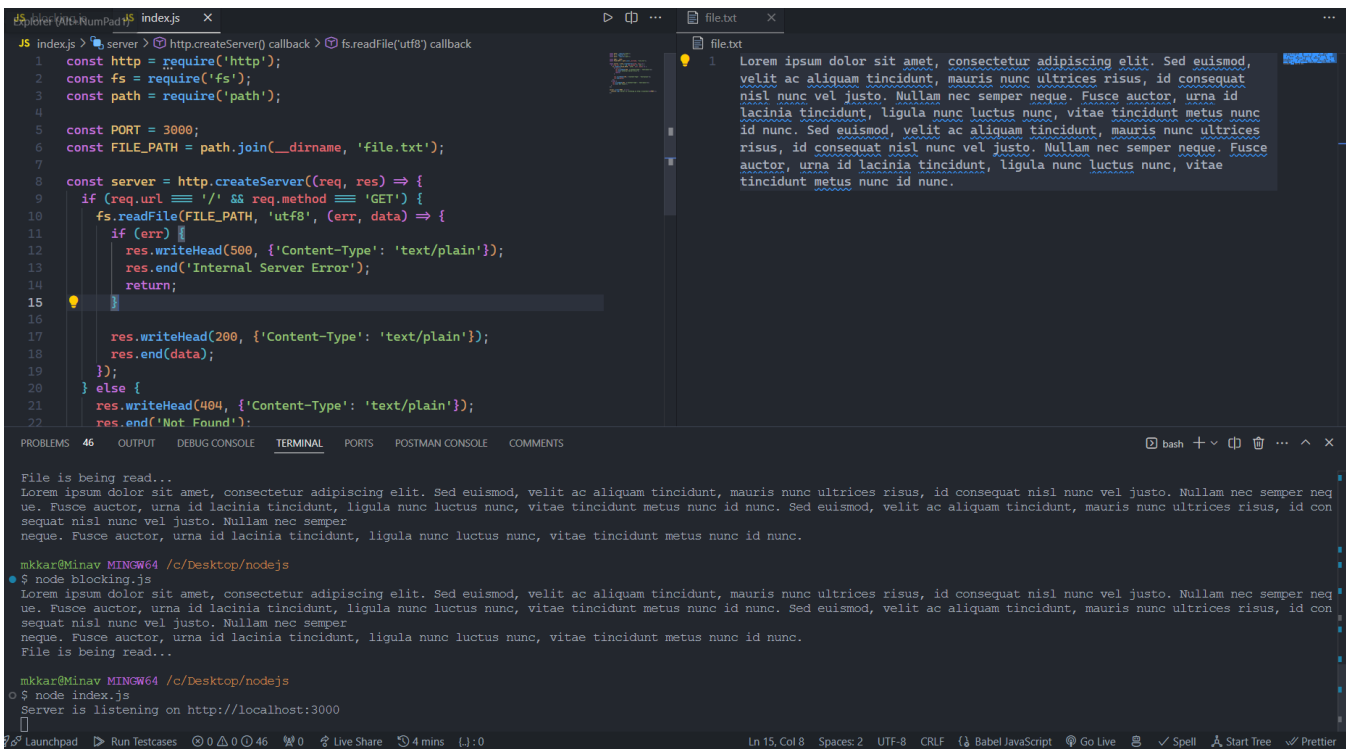


Output:



localhost:3000

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed euismod, velit ac aliquam tincidunt, mauris nunc ultrices risus, id consequat nisl nunc vel justo. Nullam nec semper neque. Fusce auctor, urna id lacinia tincidunt, ligula nunc luctus nunc, vitae tincidunt metus nunc id nunc. Sed euismod, velit ac aliquam tincidunt, mauris nunc ultrices risus, id consequat nisl nunc vel justo. Nullam nec semper neque. Fusce auctor, urna id lacinia tincidunt, ligula nunc luctus nunc, vitae tincidunt metus nunc id nunc.



```

index.js
1 const http = require('http');
2 const fs = require('fs');
3 const path = require('path');
4
5 const PORT = 3000;
6 const FILE_PATH = path.join(__dirname, 'file.txt');
7
8 const server = http.createServer((req, res) => {
9   if (req.url === '/' && req.method === 'GET') {
10     fs.readFile(FILE_PATH, 'utf8', (err, data) => {
11       if (err) {
12         res.writeHead(500, {'Content-Type': 'text/plain'});
13         res.end('Internal Server Error');
14         return;
15       }
16
17       res.writeHead(200, {'Content-Type': 'text/plain'});
18       res.end(data);
19     });
20   } else {
21     res.writeHead(404, {'Content-Type': 'text/plain'});
22     res.end('Not Found');
23   }
24 });
25
26 server.listen(PORT, () => {
27   console.log(`Server is listening on http://localhost:${PORT}`);
28 });

```

```

file.txt
1 Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed euismod, velit ac aliquam tincidunt, mauris nunc ultrices risus, id consequat nisl nunc vel justo. Nullam nec semper neque. Fusce auctor, urna id lacinia tincidunt, ligula nunc luctus nunc, vitae tincidunt metus nunc id nunc. Sed euismod, velit ac aliquam tincidunt, mauris nunc ultrices risus, id consequat nisl nunc vel justo. Nullam nec semper neque. Fusce auctor, urna id lacinia tincidunt, ligula nunc luctus nunc, vitae tincidunt metus nunc id nunc.

```

```

mkkar@Minav MINGW64 /c/Desktop/nodejs
$ node blocking.js
File is being read...
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed euismod, velit ac aliquam tincidunt, mauris nunc ultrices risus, id consequat nisl nunc vel justo. Nullam nec semper neque. Fusce auctor, urna id lacinia tincidunt, ligula nunc luctus nunc, vitae tincidunt metus nunc id nunc. Sed euismod, velit ac aliquam tincidunt, mauris nunc ultrices risus, id consequat nisl nunc vel justo. Nullam nec semper neque. Fusce auctor, urna id lacinia tincidunt, ligula nunc luctus nunc, vitae tincidunt metus nunc id nunc.
File is being read...

mkkar@Minav MINGW64 /c/Desktop/nodejs
$ node index.js
Server is listening on http://localhost:3000

```

## 2) Building your custom modules

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

Code :

```
function add(a, b) {  
  return a + b;  
}  
  
function subtract(a, b) {  
  return a - b;  
}  
  
function multiply(a, b) {  
  return a * b;  
}  
  
function divide(a, b) {  
  if (b === 0) {  
    throw new Error('Cannot divide by zero');  
  }  
  return a / b;  
}  
  
module.exports = {  
  add,  
  subtract,  
  multiply,  
  divide  
};
```

```
const readline = require('readline');  
const math = require('./math');  
  
const rl = readline.createInterface({  
  input: process.stdin,  
  output: process.stdout  
});  
  
const promptUser = () => {  
  rl.question('Enter the first number: ', (num1) => {
```



```
rl.question('Enter the second number: ', (num2) => {
  rl.question('Enter an operation (add, subtract, multiply, divide): ',
(operation) => {
  const number1 = parseFloat(num1);
  const number2 = parseFloat(num2);

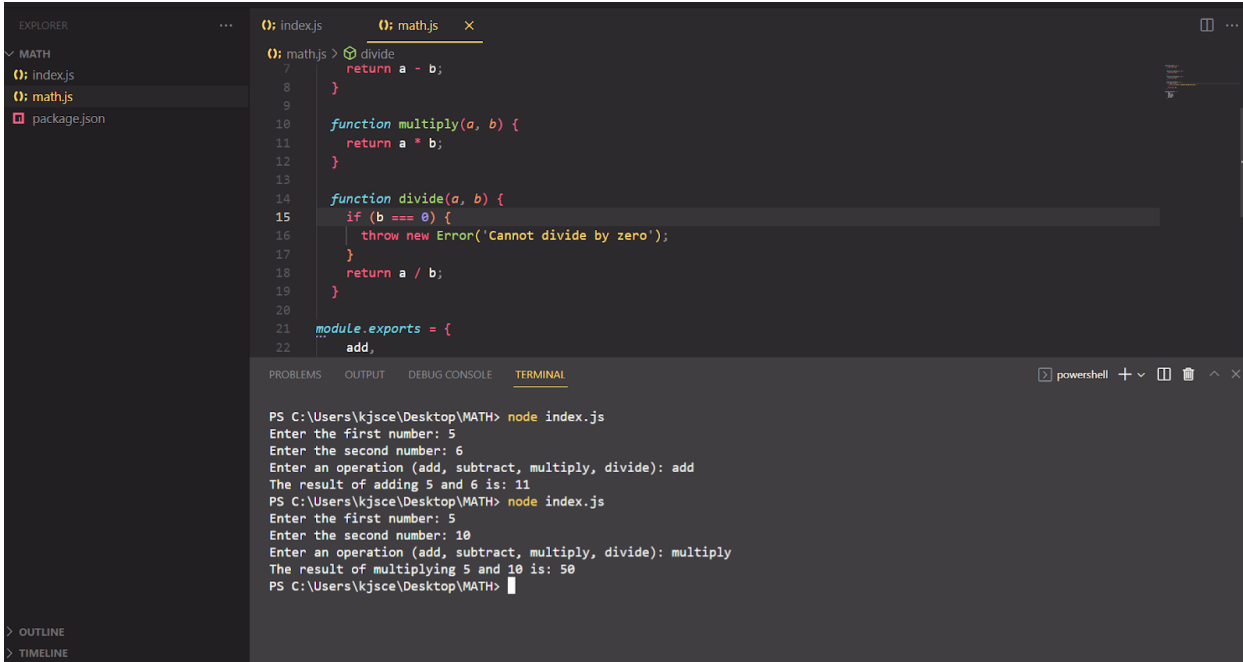
  try {
    let result;
    switch (operation.toLowerCase()) {
      case 'add':
        result = math.add(number1, number2);
        break;
      case 'subtract':
        result = math.subtract(number1, number2);
        break;
      case 'multiply':
        result = math.multiply(number1, number2);
        break;
      case 'divide':
        result = math.divide(number1, number2);
        break;
      default:
        console.log('Invalid operation. Please choose from add, subtract,
multiply, or divide.');
```

```
        rl.close();
        return;
      }
      console.log(`The result of ${operation}ing ${number1} and ${number2}
is: ${result}`);
    } catch (error) {
      console.error('Error:', error.message);
    }

    rl.close();
  });
});
});
};

promptUser();
```

Output :



The screenshot shows a Visual Studio Code editor with a file named `index.js` open. The file contains a JavaScript module with functions for addition, subtraction, multiplication, and division. The terminal window at the bottom shows the execution of the script using `node index.js`. The user is prompted to enter two numbers and an operation. The first run shows the result of adding 5 and 6 is 11. The second run shows the result of multiplying 5 and 10 is 50.

```
0: index.js 0: mathjs x
0: mathjs > divide
7   return a - b;
8   }
9
10  function multiply(a, b) {
11    return a * b;
12  }
13
14  function divide(a, b) {
15    if (b === 0) {
16      throw new Error('Cannot divide by zero');
17    }
18    return a / b;
19  }
20
21  module.exports = {
22    add,
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS C:\Users\kjsce\Desktop\MATH> node index.js  
Enter the first number: 5  
Enter the second number: 6  
Enter an operation (add, subtract, multiply, divide): add  
The result of adding 5 and 6 is: 11  
PS C:\Users\kjsce\Desktop\MATH> node index.js  
Enter the first number: 5  
Enter the second number: 10  
Enter an operation (add, subtract, multiply, divide): multiply  
The result of multiplying 5 and 10 is: 50  
PS C:\Users\kjsce\Desktop\MATH>

### 3) 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

Code :

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

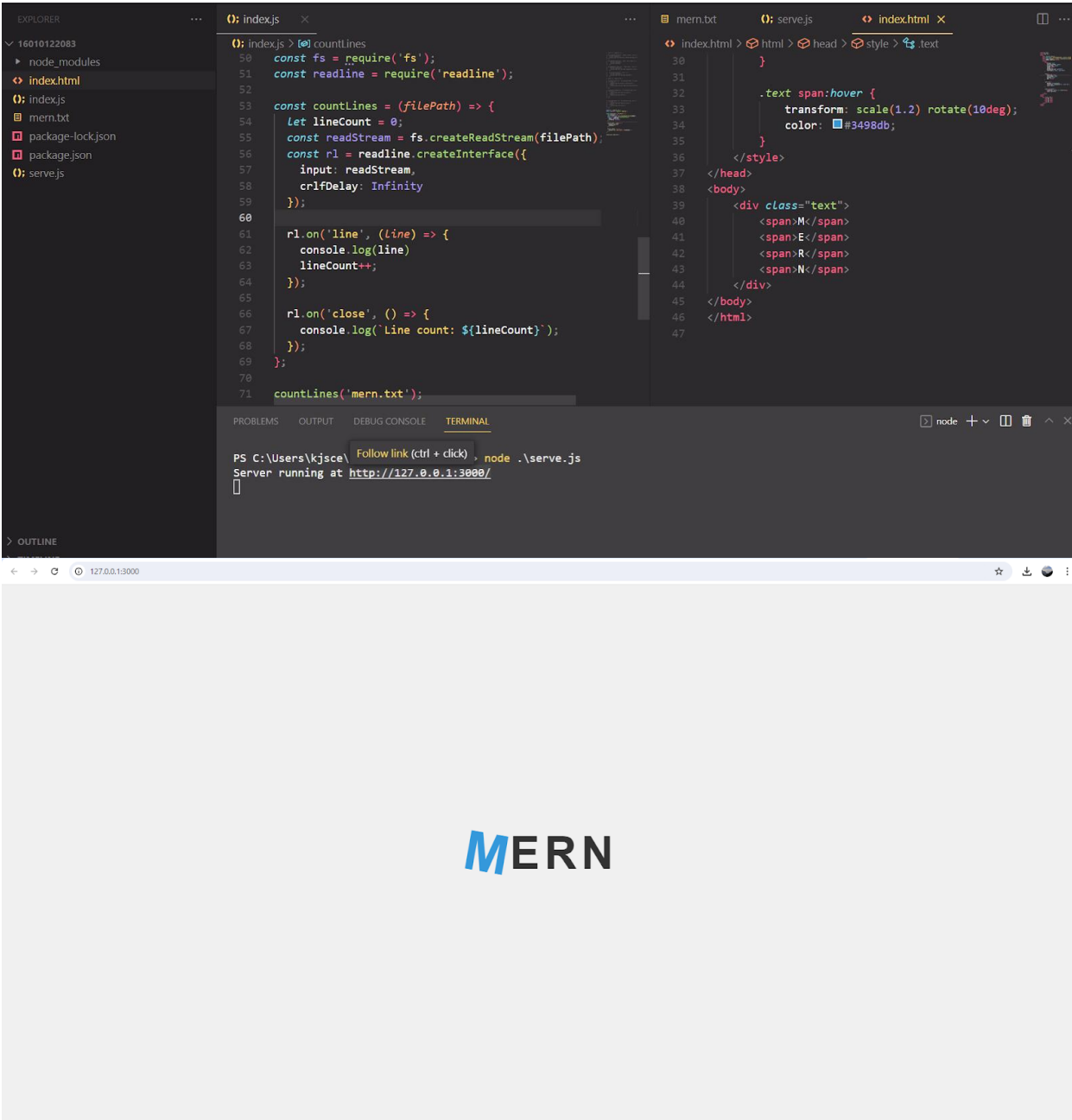
const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  if (req.url === '/') {
    serveFile('index.html', 'text/html', res);
  } else if (req.url === '/data') {
    serveFile('data.json', 'application/json', res);
  } else {
    res.statusCode = 404;
    res.setHeader('Content-Type', 'text/plain');
    res.end('404 Not Found');
  }
});

function serveFile(filePath, contentType, res) {
  const fullPath = path.join(__dirname, filePath);
  fs.readFile(fullPath, (err, data) => {
    if (err) {
      res.statusCode = 500;
      res.setHeader('Content-Type', 'text/plain');
      res.end('500 Internal Server Error');
      return;
    }
    res.statusCode = 200;
    res.setHeader('Content-Type', contentType);
    res.end(data);
  });
}

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

Output :



```

0: index.js
50 const fs = require('fs');
51 const readline = require('readline');
52
53 const countLines = (filePath) => {
54   let lineCount = 0;
55   const readStream = fs.createReadStream(filePath);
56   const rl = readline.createInterface({
57     input: readStream,
58     crlfDelay: Infinity
59   });
60
61   rl.on('line', (line) => {
62     console.log(line);
63     lineCount++;
64   });
65
66   rl.on('close', () => {
67     console.log(`Line count: ${lineCount}`);
68   });
69 };
70
71 countLines('mern.txt');
  
```

```

0: serve.js
30
31
32 .text span:hover {
33   transform: scale(1.2) rotate(10deg);
34   color: #3498db;
35 }
36
37 </head>
38 <body>
39   <div class="text">
40     <span>M</span>
41     <span>E</span>
42     <span>R</span>
43     <span>N</span>
44   </div>
45 </body>
46 </html>
47
  
```

```

PS C:\Users\kjsce\ Follow link (ctrl + click) > node .\serve.js
Server running at http://127.0.0.1:3000/
  
```

MERN

```
127.0.0.1:3000/data
Pretty print ☐
{
  "message": "Hello, this is JSON data!",
  "status": "success"
}
```

#### 4) Blocking and Non Blocking

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

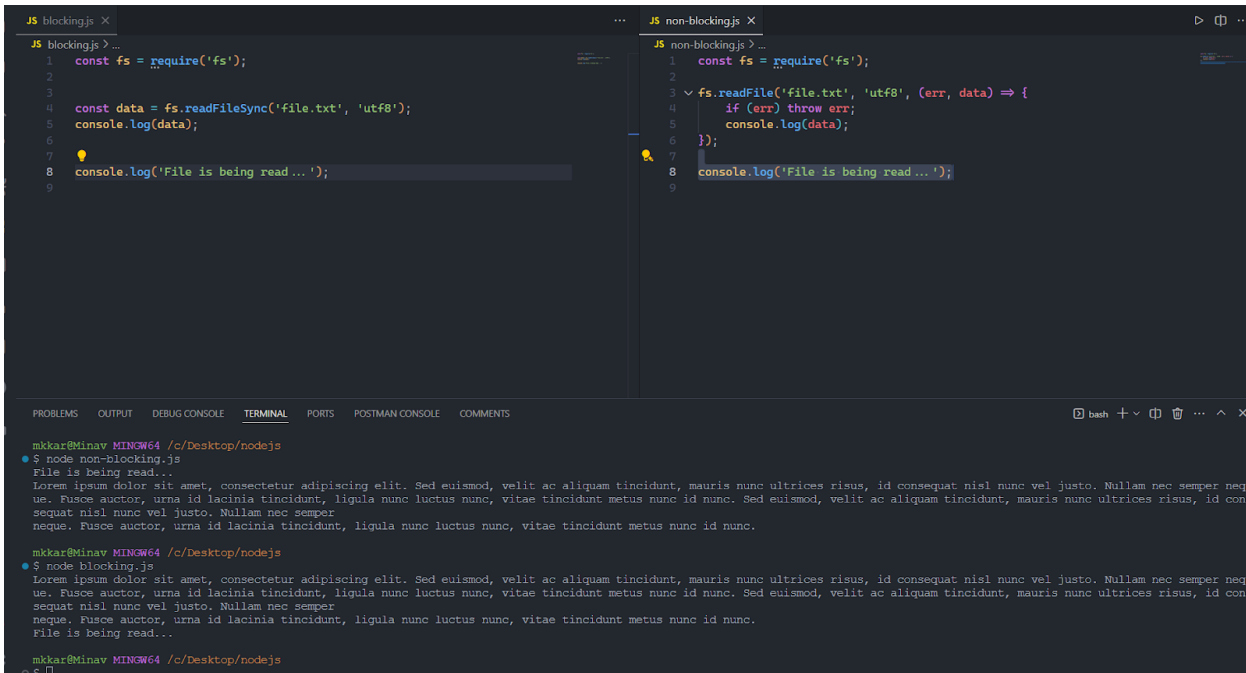
const data = fs.readFileSync('file.txt', 'utf8');
console.log(data);

console.log('File is being read...');
```

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

fs.readFile('file.txt', 'utf8', (err, data) => {
  if (err) throw err;
  console.log(data);
});

console.log('File is being read...');
```



```
JS blocking.js X
JS blocking.js > ...
1 const fs = require('fs');
2
3
4 const data = fs.readFileSync('file.txt', 'utf8');
5 console.log(data);
6
7
8 console.log('File is being read...');
9

JS non-blocking.js X
JS non-blocking.js > ...
1 const fs = require('fs');
2
3 fs.readFile('file.txt', 'utf8', (err, data) => {
4   if (err) throw err;
5   console.log(data);
6 });
7
8 console.log('File is being read...');
9
```

```
mkkar@Minav MINGW64 /c/Desktop/nodejs
$ node non-blocking.js
File is being read...
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed euismod, velit ac aliquam tincidunt, mauris nunc ultrices risus, id consequat nisl nunc vel justo. Nullam nec semper neque. Fusce auctor, urna id lacinia tincidunt, ligula nunc luctus nunc, vitae tincidunt metus nunc id nunc. Sed euismod, velit ac aliquam tincidunt, mauris nunc ultrices risus, id consequat nisl nunc vel justo. Nullam nec semper neque. Fusce auctor, urna id lacinia tincidunt, ligula nunc luctus nunc, vitae tincidunt metus nunc id nunc.

mkkar@Minav MINGW64 /c/Desktop/nodejs
$ node blocking.js
File is being read...
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed euismod, velit ac aliquam tincidunt, mauris nunc ultrices risus, id consequat nisl nunc vel justo. Nullam nec semper neque. Fusce auctor, urna id lacinia tincidunt, ligula nunc luctus nunc, vitae tincidunt metus nunc id nunc.

mkkar@Minav MINGW64 /c/Desktop/nodejs
$
```



**Conclusion:**

We learned file handling and browser routing with nodejs and its implementation