**Practical No.: 09**

**Aim**: Perform following operations in Node.js

1. Reading a file
2. Writing a file
3. **Reading a file**

create sample.txt file with some content

A screenshot of a computer

Description automatically generated

1. Synchronously reads the entire contents of a file.

**Read.js:**

const fs = require("fs")

const a = fs.readFileSync('./sample.txt', "utf-8")

console.log(a)

**Output:ge**

A black screen with white text

Description automatically generated

1. Asynchronously reads the entire contents of a file.

**Async.js:**

const fs = require("fs")

fs.readFile("./sample.txt", "utf-8",

(error, data)=>{

if(error){

throw new Error('Error reading file!')

}

console.log(data)

}

)

**Output:**

**A black background with white text

Description automatically generated**

1. Read a file content in Async function.

**Async\_fun.js:**

const fs = require("fs").promises;

async function readFile(){

try{

const data = await fs.readFile("sample.txt", "utf-8");

console.log(data);

}

catch(err){

console.log(err);

}

}

readFile()

**Output:**

A black background with white text

Description automatically generated

1. **Writing a file**
2. Create a file example.txt
3. Add some content to it. Read the file.
4. Append data to the file
5. Rename the file
6. Read the file again after appending
7. Delete the file

const fs = require('fs/promises')

const writeFunct = async () =>{

try{

const d= await fs.readFile('example.txt', 'utf-8');

console.log(d)

await fs.writeFile('example.txt', 'Writing in a file', "utf-8")

await fs.appendFile('example.txt', '\n data append via node.js', "utf-8")

await fs.rename('example.txt','NewWrite.txt')

const data = await fs.readFile('NewWrite.txt', "utf-8")

console.log(data)

}

catch(err){

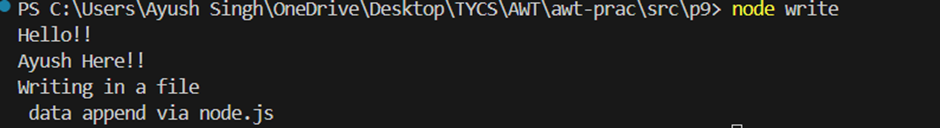
throw err

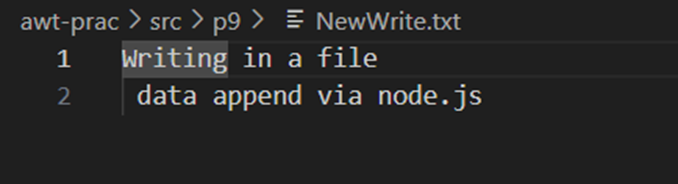
}

}

writeFunct()

**Output:**

****

****

1. Create a Node.js program that allows users to input text via the command line. Once the user enters the text, the program should save the input into a file named text.txt. Additionally, the program should emit a custom event indicating that the text is ready for processing.

const fs = require('fs');

const readline = require('readline');

const EventEmitter = require('events');

const eventEmitter = new EventEmitter();

const rl = readline.createInterface({

input: process.stdin,

output: process.stdout

});

function handleInput(input) {

eventEmitter.emit('textReady', input);

}

eventEmitter.on('textReady', (text) => {

console.log(`Custom event fired: Text is ready - ${text}`);

fs.writeFile('text.txt', text, (err) => {

if (err) throw err;

console.log('Text has been saved to text.txt');

rl.close();

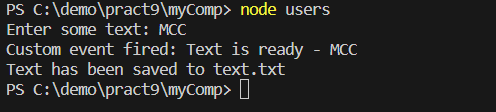
});

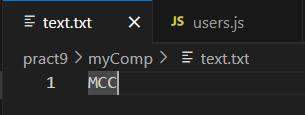
});

rl.question('Enter some text: ', (text) => {

handleInput(text);

});





1. Create a simple Node.js program that creates an HTTP server and handles requests to it.

const http = require('http');

const hostname = '127.0.0.1';

const port = 4000;

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

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

if (req.url === '/hello') {

res.end('Hello World!\n');

} else if (req.url === '/about') {

res.end('This is the about page.\n');

} else {

res.end('Node.js server!\n');

}

});

server.listen(port, hostname, () => {

console.log(`Server running at http://${hostname}:${port}/`);

});

**Output:**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**