Web Technologies Module 3 Assignment 2

Atman Shastri, FYMCA-C, 182

1. Aim:
   To know about how to get directory name, base name and extension name of a file

   Theory:
   To be able to manipulate paths in order to write a program

   Code:

```
const { join, resolve } = require('path');
var path = require('path');

const filepath = 'C:/Users/admin/Desktop/Batch C Roll No 182/WT/Module 3/data.txt'

console.log("Directory Name: " + path.dirname(filepath));
console.log("Base Name: " + path.basename(filepath));
console.log("Extension Type: " + path.extname(filepath));

var name = 'mca'
console.log(path.join("User Defined Path: ","/","users",name,'data.txt'));
console.log("Actual Path: " + path.resolve("data.txt"));
```

Output:

```
PS F:\D\MCA\Assignments & Backup Files\wt\Module 3> node .\1_filepath.js
Directory Name: C:/Users/admin/Desktop/Batch C Roll No 182/WT/Module 3
Base Name: data.txt
Extension Type: .txt
User Defined Path: \users\mca\data.txt
Actual Path: F:\D\MCA\Assignments & Backup Files\wt\Module 3\data.txt
PS F:\D\MCA\Assignments & Backup Files\wt\Module 3>
```

2. Aim:
   To create an asynchronous function using the await keyword

   Theory: To understand the asynchronous nature and understand when to use the await keyword in the program.

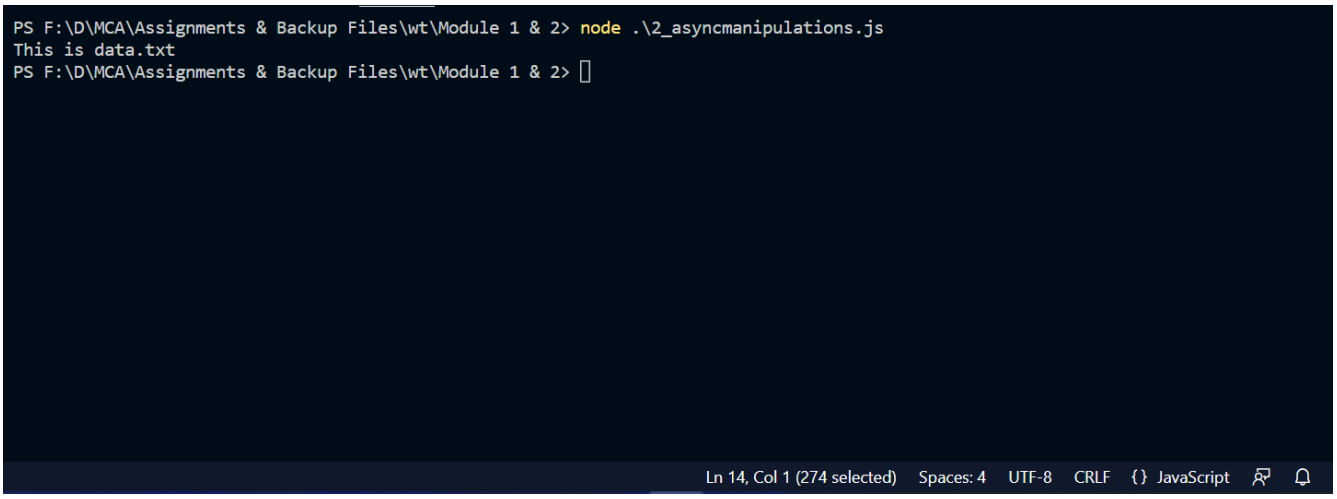
   Code:

```
var fs=require('fs').promises;

async function readFile(FilePath) {
    try {
    var data = await fs.readFile(FilePath);
    console.log(data.toString());
    }
    catch(error) {
    console.log("Error Occurred while reading");
    }
}

readFile("data.txt");
```

Output:



3. Aim:
   To write data in a csv file using node js program

   Theory:
   Using Promise based API to implement asynchronous write operations

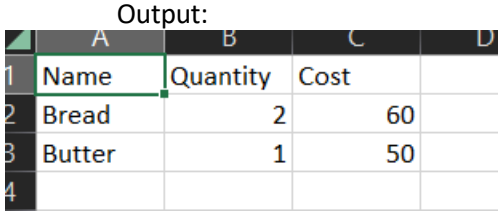
   Code:

```
var fs=require('fs').promises;

async function writetocsv(){
    try{
        const csvheader = "Name,Quantity,Cost";
        await fs.writeFile("Groceries.csv",csvheader);
    } catch(error) {
        console.log("Error Occured"+error);
    }
}
async function additems(Name,Quantity,Cost){
    try{
        var csvline= `\n${Name},${Quantity},${Cost}`;
        await fs.writeFile("Groceries.csv",csvline,{flag:'a'})
    }catch(error){
        console.log("Error Occurred while appending"+error);
    }
}

writetocsv();
additems("Bread",2,60);
additems("Butter",1,50);
```

   Output:



	A	B	C	D
1	Name	Quantity	Cost	
2	Bread	2	60	
3	Butter	1	50	
4				

4. Aim:
   To read data from a file using a buffer

   Theory:
   To learn how to create and read a buffer in order to program better

   Code:

```
var fs = require('fs');
fs.open("data.txt",'r',(err,fd)=>{
    if (err) {
        console.log("Error Occurred"+err);
    }
    else{
        var buffer = new Buffer.alloc(1024);
        fs.read(fd,buffer,0,buffer.length,0,(err,bytes)=>{
            console.log(buffer.slice(0,bytes).toString())
        });
    }
    fs.close(fd);
});
```

   Output:

```
PS F:\D\MCA\Assignments & Backup Files\WT\Module 3> node .\2_Buffer_read.js
Welcome to my website!!!
PS F:\D\MCA\Assignments & Backup Files\WT\Module 3>
```

Ln 12, Col 18 Spaces: 4 UTF-8 CRLF {} JavaScript

5. Aim:
   To write data to a file using a buffer
   Theory:
   To learn how to create and write to a buffer in order to program better

   Code:

```
var fs = require('fs');
fs.open("data.txt",'a',(err,fd)=>{
    if (err) {
        console.log("Error Occurred"+err);
    }
    else{
        var buffer = new Buffer.alloc(1024);
        buffer.write("New Data is here");
        fs.write(fd,buffer,0,buffer.length,null,(err,bytes)=>{
            console.log("Wrote " + bytes + " bytes")
```

```
        });
    }
    fs.close(fd);
})
```

Output:

```
PS F:\D\MCA\Assignments & Backup Files\WT\Module 3> node .\2_Buffer_write.js
Wrote 1024 bytes
PS F:\D\MCA\Assignments & Backup Files\WT\Module 3>
```

Ln 14, Col 4 (388 selected) Spaces: 4 UTF-8 CRLF {} JavaScript

6. Aim: Read and write data from a file using stream
   Theory: To learn the implementation of streams in node js
   Code:

```
var fs = require('fs');
var readstream = fs.createReadStream("data.txt");
var writestream =fs.createWriteStream("writefile.txt")
readstream.on("data",function(filedata){
    writestream.write(filedata);
    console.log(filedata.toString())
});
```

Output:

```
PS F:\D\MCA\Assignments & Backup Files\WT\Module 3> node .\2_readstream.js
Welcome to my website!!!
PS F:\D\MCA\Assignments & Backup Files\WT\Module 3>
```

Ln 13, Col 4 Spaces: 4 UTF-8 CRLF {} JavaScript

7. Aim:
   To pipe a text file as a response

   Theory:
   Understanding to pipe text files in order to achieve same output with less code

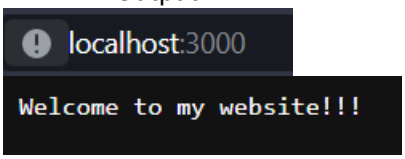
   Code:

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

var server = http.createServer((req,res)=> {
    res.writeHead(200,{'Content-Type':'text/plain'});
    var datastream = fs.createReadStream("data.txt");
    datastream.pipe(res);
});

server.listen(3000);
```

   Output:



8. Aim:
   To pipe a html file as a response

   Theory:
   Understanding to pipe files other than .txt files in order to achieve same output with less code

   Code:

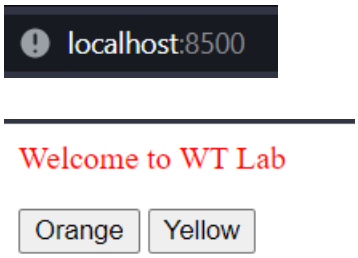
```
var http = require('http');
var fs = require('fs');

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

    res.writeHead(200,{'Content-Type': 'text/html'});
    var readstream= fs.createReadStream("module_3_index.html");
    readstream.pipe(res);
});

server.listen(8500);
```

   Output:



9. Aim: To close server after a set timeout

   Theory:
   To implement a timeout function in order to close a server that is not in use anymore within the execution of program

   Code:

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

var server = http.createServer((req,res)=> {
   //
});

server.listen(9000);

setTimeout(()=> {
    server.close();
    server.unref();
```

```
},10000);
```

Output:

```
PS F:\D\MCA\Assignments & Backup Files\WT\Module 3> node .\4_close_server.js
PS F:\D\MCA\Assignments & Backup Files\WT\Module 3>
```

Ln 12, Col 8 Spaces: 4 UTF-8 CRLF {} JavaScript

10. Aim: To read data from a file and send it as response when “data” event is triggered

Theory:
To output data from file when “data” event is triggered

Code:

```
var fs = require('fs');
var http = require('http');
var readstream = fs.createReadStream("data.txt");
readstream.on("data",function(filedata){
    console.log(filedata.toString())
});
```

Output:

```
PS F:\D\MCA\Assignments & Backup Files\WT\Module 3> node .\2_readstream.js
Welcome to my website!!!
PS F:\D\MCA\Assignments & Backup Files\WT\Module 3>
```

Ln 1, Col 1 (188 selected) Spaces: 4 UTF-8 CRLF {} JavaScript

11. Aim: To redirect user to different pages based on conditions in url

Theory:
To redirect user to dashboard if url has “/dashboard” or else respond with 404 not found message

Code:

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

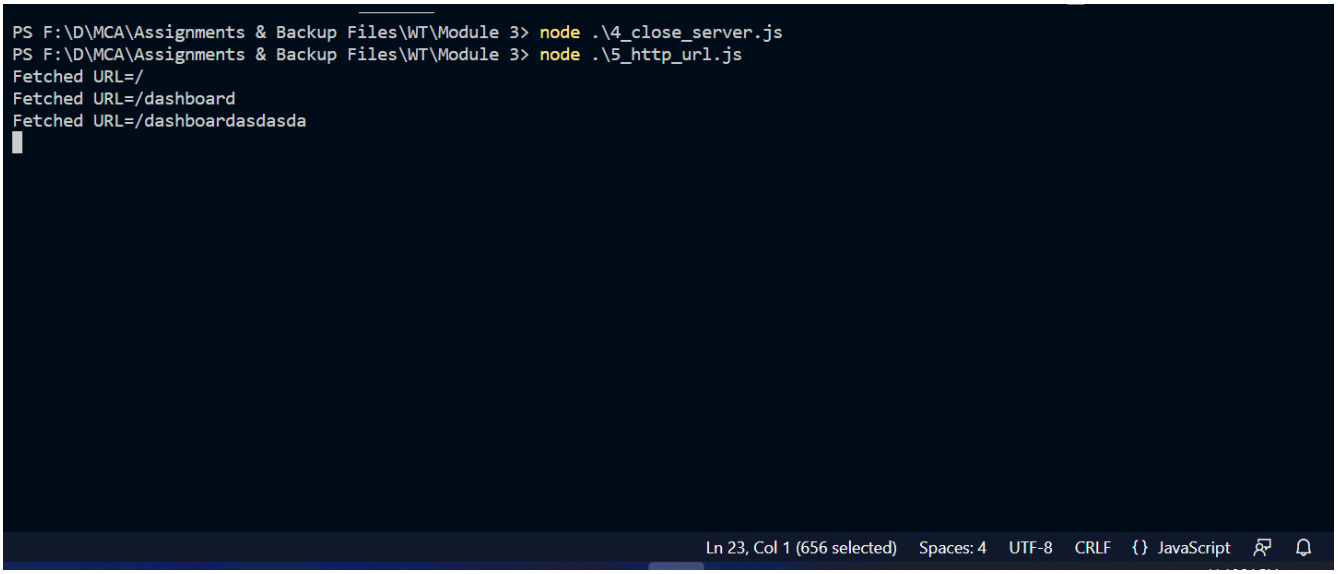
```
var server = http.createServer();

server.on("request",(req,res)=>{
        var url = req.url;
        console.log("Fetched URL=" + url);
        if(url==("/dashboard")) {
            res.writeHead(200,{'Content-Type' : 'text/html'});
            //res.write("This is my dashboard");
            var DashboardRead = fs.createReadStream("dashboard.html");
            DashboardRead.pipe(res);

        } else {
           // res.writeHead(404,{'Content-Type': 'text/plain'})
            res.write("404 \n Page not found");
            res.end();
        }
});

server.listen(5000);
```

Output:



12. Aim: Take input data from user using “post” and display the data entered by the user

Theory:
To display form, take input and show data to user after input

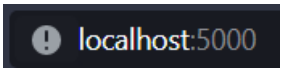
Code:

```
var http = require('http');
var fs = require('fs');
var qs = require('querystring');
var mysql = require('mysql');

var con = mysql.createConnection({
    host: "localhost",
    port: "3308",
    user: "root",
    password: "",
    database: "college"
});
var server = http.createServer((req, res) => {
    var body = "";
    if (req.method == 'GET') {
        res.writeHead(200, { 'Content-Type': 'text/html' })
        fs.createReadStream("register.html").pipe(res);
    }
    else if (req.method == 'POST') {
        var formdata = "";
        req.on("data", (chunk) => {
            formdata += chunk;
            var data = qs.parse(formdata);
            body = "\n Name: " + data.sname + "\n Phone no: " + data.scontact + "\n
Address: " + data.saddress;
            con.connect((err) => {
                if (err) throw err;
                var sql = "Insert into student(Name,Contact,Address) values('" +
data.sname + "','" + data.scontact + "','" + data.saddress + "')";
                con.query(sql, (error, result) => {
                    if (error) throw error;
                    console.log(result);
                });
            });
        });
    }
    req.on("end", () => {
        res.writeHead(200, { 'Content-Type': 'text/html' });
        res.end(body);
    })
});

server.listen(5000);
```

Output:



Name: as

Contact No: 234234234

Address: asda

Submit

Name: as Phone no: 234234234 Address: asda