Slip 5

Q1: Node.js Server to Read content of a File and Display on Browser

■ fileServer.js

```
const http=require('http');
const fs=require('fs');
const path=require('path');
const filePath=path.join(__dirname,'sample.txt');

const server=http.createServer((req,res)=>{
  fs.readFile(filePath,'utf8',(err,data)=>{
    if(err){
      res.writeHead(500,{'Content-Type':'text/plain'});
      return res.end('Error reading file.');
    }
    res.writeHead(200,{'Content-Type':'text/plain'});
    res.end(data);
  });
});

server.listen(3000,()=>console.log("Server running at http://localhost:3000/"));
```

Q2: Node.js Server to Write content in a File and Display result on Browser

■ writeFileServer.js

```
const http=require('http');
const fs=require('fs');
const path=require('path');
const filePath=path.join(__dirname,'output.txt');

const server=http.createServer((req,res)=>{
  const content="Hello! Content written by Node.js server.\nTimestamp: "+new Date();
  fs.writeFile(filePath,content,'utf8',(err)=>{
    if(err){
      res.writeHead(500,{'Content-Type':'text/plain'});
      return res.end('Error writing file.');
    }
    res.writeHead(200,{'Content-Type':'text/html'});
    res.end(`<h1>File written successfully!</h1><p>Content saved to ${filePath}</p>`);
  });
});

server.listen(3000,()=>console.log("Server running at http://localhost:3000/"));
```

Slip 3

Q1: Node.js user defined module MyDateModule.js & access it

■ MyDateModule.js

```
// MyDateModule.js
function getCurrentDate(){return new Date().toISOString().split('T')[0];}
function getCurrentTime(){return new Date().toTimeString().split(' ')[0];}
function getDayName(){return ['Sunday','Monday','Tuesday','Wednesday','Thursday','Friday','Satur
module.exports={getCurrentDate,getCurrentTime,getDayName};
```

■ app.js

```
const dateModule=require('./MyDateModule');
console.log("Current Date:",dateModule.getCurrentDate());
console.log("Current Time:",dateModule.getCurrentTime());
console.log("Day:",dateModule.getDayName());
```

Q2: Node.js module to convert string to uppercase & reverse

■ myStringUpperReverseModule.js

```
function toUpper(str){return str.toUpperCase();}
function reverse(str){return str.split('').reverse().join('');}
module.exports={toUpper,reverse};
```

■ app.js

```
const str=require('./myStringUpperReverseModule');
const input="hello world";
console.log("Original:",input);
console.log("Upper:",str.toUpper(input));
console.log("Reverse:",str.reverse(input));
```

Slip 4

Q1: Node.js Server to display Good Afternoon message

■ afternoonServer.js

```
const http=require('http');
const server=http.createServer((req,res)=>{
  res.writeHead(200,{"Content-Type":"text/html"});
  res.end("<h1 style='color:orange;text-align:center;'>Good Afternoon!</h1>");
});
server.listen(3000,()=>console.log("Server running at http://localhost:3000/"));
```

Q2: Node.js Script to demonstrate Chalk NPM

■ chalkDemo.js

```
// Install first: npm init -y && npm install chalk
const chalk=require('chalk');
console.log(chalk.green('■ Success message'));
console.log(chalk.red('■ Error message'));
console.log(chalk.blue('■■ Info message'));
console.log(chalk.yellow('■■ Warning message'));
console.log(chalk.bold.underline.magenta('Bold, underlined magenta'));
console.log(chalk.black.bgCyan('Cyan background text'));
const username='Alice';
console.log(chalk`Hello {bold.green ${username}}! Welcome to {bgYellow.black Node.js}`);
```

Slip 1

Q1: JavaScript Program to display Timer

■ timer.html

```
<!DOCTYPE html>
<html>
<head><title>Simple Timer</title></head>
<body>
  <h1>Timer: <span id="timer">0</span></h1>
  <script>
    let seconds = 0;
    function updateTimer() {
      document.getElementById('timer').textContent = seconds;
      seconds++;
    }
    setInterval(updateTimer, 1000);
  </script>
</body>
</html>
```

Q2: Node Server Script to add two numbers

■ add-server.js

```
const http = require("http");
const url = require("url");

const server = http.createServer((req, res) => {
  const query = url.parse(req.url, true).query;
  const num1 = parseFloat(query.num1);
  const num2 = parseFloat(query.num2);

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

  if (!isNaN(num1) && !isNaN(num2)) {
    res.end(`Sum: ${num1 + num2}`);
  } else {
    res.end("Go to http://localhost:3000/?num1=10&num2=20");
  }
});

server.listen(3000, () => {
  console.log("■ Server running at http://localhost:3000/");
});
```

Slip 2

Q1: JavaScript Validation of Username and Age

■ validate.html

```
<!DOCTYPE html>
<html>
<head><title>User Validation</title></head>
<body>
  <form onsubmit="return validateForm()">
    <label>Username:</label><input type="text" id="username" required><br>
    <label>Age:</label><input type="number" id="age" required><br>
    <div id="error" style="color:red;"></div>
    <button type="submit">Submit</button>
  </form>
  <script>
    function validateForm(){
      const u=document.getElementById('username').value.trim();
      const a=parseInt(document.getElementById('age').value.trim());
      const err=document.getElementById('error'); err.innerHTML="";
      if(u.length<3){err.textContent="Username min 3 chars"; return false;}
      if(isNaN(a)||a<1||a>120){err.textContent="Enter valid age 1-120"; return false;}
      alert("Validation successful: "+u+" ("+a+")");
      return true;
    }
  </script>
</body>
</html>
```

Q2: Node Module (sayHello & fact)

■ myModule.js

```
function sayHello(name){return `Hello, ${name}!`;}
function fact(n){let r=1;for(let i=2;i<=n;i++){r*=i;}return r;}
module.exports={sayHello,fact};
```

■ app.js

```
const my=require('./myModule');
console.log(my.sayHello('Alice'));
console.log("Factorial of 5:",my.fact(5));
```