Slip1 :

1. Java Script Program to display Timer [timer.js](http://timer.js) . [10 Marks ]
2. Write a Simple Node Server Script to find addition of two number

# [20 Marks]

Slip2 :

# Write a Java Script Program to Validation of User Name and age [10 Marks]

# Create a NODE JS module (sayHello & fact) & use it to display factorial and Hello Message.

# [20 Marks]

Slip3 :

1. Create Node JS user defined module MyDateModule.js & access it

[10 Marks]

1. Create NODE JS user defined module to convert a string to uppercase & reverse (myStringUpperReverseModule.js) [20 Marks]

Slip4:

1. Write a NODE JS Server to Display Good After Noon Message

# [10 Marks]

1. Write a NODE JS Script to demonstrate Chalk NPM [20 Marks]

Slip5 :

1. Create a NODE JS Server to Read content of the File and Display on Browser

# [10 Marks]

1. Create a NODE JS Server to Write content in the File and Display result on Browser

# [20 Marks]

Slip 6 :

# Write a NODE JS Server to Display Good After Noon Message

1. Create a NODE JS Server to Write content in the File and Display result on Browser

Slip7 :

# Java Script Program to display Timer timer.js [10 Marks]

# Create a NODE JS module (sayHello & fact) & use it to display factorial and Hello Message.

Slip8 :

1. Create Node JS user defined module MyDateModule.js & access it
2. Implement CRUD (Create, Read, Update, Delete) operations using Node.js and Express.

Slip 9 :

# Write a NODE JS Server to wish(Good Morning/After Noon) according to Server Time

1. Build a Node.js server that accepts file uploads from a simple HTML form.

Slip10 :

# Write a Java Script Program to Validation of User Name and password

1. Create a NODE JS Server to Write content in the File and Display result on Browser

**Slip1 :**

1. **Java Script Program to display Timer timer.js**

[**timer.js**](http://timer.js)

**let seconds = 0;**

**function updateTimer() {**

**document.getElementById('timer').textContent = seconds;**

**seconds++;**

**}**

**setInterval(updateTimer, 1000);**

**timer.html**

**<!DOCTYPE html>**

**<html >**

**<head>**

**<title>Simple Timer</title>**

**</head>**

**<body>**

**<h1>Timer: <span id="timer">0</span></h1>**

**<script src="timer.js"></script>**

**</body>**

**</html>**

**2) Write a Simple Node Server Script to find addition of two number**

**!-- index.html -->**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<title>Add Two Numbers</title>**

**</head>**

**<body>**

**<h1>Addition Form</h1>**

**<form action="http://localhost:3000" method="get">**

**<label for="num1">Number 1:</label>**

**<input type="number" name="num1" id="num1" required>**

**<br><br>**

**<label for="num2">Number 2:</label>**

**<input type="number" name="num2" id="num2" required>**

**<br><br>**

**<button type="submit">Add</button>**

**</form>**

**</body>**

**</html>**

**// add-server.js**

**const http = require('http');**

**const url = require('url');**

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

**const queryObject = url.parse(req.url, true).query;**

**const num1 = parseFloat(queryObject.num1);**

**const num2 = parseFloat(queryObject.num2);**

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

**if (!isNaN(num1) && !isNaN(num2)) {**

**const sum = num1 + num2;**

**res.end(`Sum of ${num1} and ${num2} is ${sum}`);**

**} else {**

**res.end('Please provide valid numbers using ?num1=VALUE&num2=VALUE');**

**}**

**});**

**server.listen(3000, () => {**

**console.log('Server running at http://localhost:3000/');**

**});**

### **How to Run**

1. **Save the file as add-server.js**
2. **Open a terminal and run:**

**node** [**add-server.js**](http://add-server.js)

**In your browser, go to:**

**http://localhost:3000/**

**Sliup2 :**

# **Write a Java Script Program to Validation of User Name and age**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>User Validation Form</title>

<script>

function validateForm() {

const username = document.getElementById('username').value.trim();

const age = document.getElementById('age').value.trim();

const errorDiv = document.getElementById('error');

errorDiv.innerHTML = '';

// Validate username

if (username.length < 3) {

errorDiv.textContent = 'Username must be at least 3 characters long.';

return false;

}

// Validate age

const ageNum = parseInt(age);

if (isNaN(ageNum) || ageNum < 1 || ageNum > 120) {

errorDiv.textContent = 'Please enter a valid age between 1 and 120.';

return false;

}

alert('Validation successful!\nUsername: ' + username + '\nAge: ' + ageNum);

return true;

}

</script>

</head>

<body>

<h1>User Form</h1>

<form onsubmit="return validateForm()">

<label for="username">Username:</label>

<input type="text" id="username" required><br><br>

<label for="age">Age:</label>

<input type="number" id="age" required><br><br>

<div id="error" style="color:red;"></div><br>

<button type="submit">Submit</button>

</form>

</body>

</html>

# **2) Create a NODE JS module (sayHello & fact) & use it to display factorial and Hello Message.**

## **✅ Step 1: Create the Module File : myModule.js**

**// Function to return a hello message**

**function sayHello(name) {**

**return `Hello, ${name}! Welcome to Node.js.`;**

**}**

**// Function to calculate factorial**

**function fact(n) {**

**if (n < 0) return 'Factorial not defined for negative numbers';**

**if (n === 0 || n === 1) return 1;**

**let result = 1;**

**for (let i = 2; i <= n; i++) {**

**result \*= i;**

**}**

**return result;**

**}**

**// Exporting functions**

**module.exports = {**

**sayHello,**

**fact**

**};**

## **✅ Step 2: Create the Main File to Use the Module : app.js**

**// Importing the custom module**

**const myModule = require('./myModule');**

**// Use sayHello**

**const helloMessage = myModule.sayHello('Alice');**

**console.log(helloMessage); // Output: Hello, Alice! Welcome to Node.js.**

**// Use fact**

**const number = 5;**

**const factorial = myModule.fact(number);**

**console.log(`Factorial of ${number} is: ${factorial}`);**

## **▶️ How to Run**

1. **Save both files (myModule.js and app.js) in the same folder.**
2. **Open a terminal in that folder.  
   Run the app: node app.js**

**Slip 3 👍**

1. **Create Node JS user defined module MyDateModule.js & access it**

## **✅ Step 1: Create the Module File**

### **📄 MyDateModule.js**

**// MyDateModule.js**

**// Get current date in YYYY-MM-DD format**

**function getCurrentDate() {**

**const now = new Date();**

**return now.toISOString().split('T')[0]; // YYYY-MM-DD**

**}**

**// Get current time in HH:MM:SS format**

**function getCurrentTime() {**

**const now = new Date();**

**return now.toTimeString().split(' ')[0]; // HH:MM:SS**

**}**

**// Get current day name (e.g., Monday)**

**function getDayName() {**

**const now = new Date();**

**const days = ['Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday'];**

**return days[now.getDay()];**

**}**

**// Export the functions**

**module.exports = {**

**getCurrentDate,**

**getCurrentTime,**

**getDayName**

**};**

## **✅ Step 2: Access the Module**

### **📄 app.js**

**// app.js**

**// Import user-defined module**

**const dateModule = require('./MyDateModule');**

**// Use the functions**

**console.log('Current Date:', dateModule.getCurrentDate());**

**console.log('Current Time:', dateModule.getCurrentTime());**

**console.log('Today is:', dateModule.getDayName());**

## **▶️ How to Run**

1. **Save both files (MyDateModule.js and app.js) in the same directory.**
2. **Open terminal/command prompt in that directory.**
3. **Run:  
     
    node** [**app.js**](http://app.js)

**2) Create NODE JS user defined module to convert a string to uppercase & reverse** ([**myStringUpperReverseModule.js**](http://mystringupperreversemodule.js))

## **✅ Step 1: Create the Module myStringUpperReverseModule.js**

// myStringUpperReverseModule.js

// Function to convert string to uppercase

function toUpper(str) {

return str.toUpperCase();

}

// Function to reverse a string

function reverse(str) {

return str.split('').reverse().join('');

}

// Export the functions

module.exports = {

toUpper,

reverse

};

## **✅ Step 2: Use the Module : app.js**

// app.js

// Import your custom module

const strUtils = require('./myStringUpperReverseModule');

// Example string

const input = 'hello world';

console.log('Original String:', input);

console.log('Uppercase:', strUtils.toUpper(input));

console.log('Reversed:', strUtils.reverse(input));

## **▶️ How to Run**

1. Save both files (myStringUpperReverseModule.js and app.js) in the **same folder**.
2. Open a terminal in that folder.
3. Run the app using:  
     
    node [app.js](http://app.js)

Slip4 :

1. Write a NODE JS Server to Display Good After Noon Message

## **✅ 1. Node.js Server: afternoonServer.js**

// afternoonServer.js

const http = require('http');

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

// Set content type to HTML

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

// Send Good Afternoon message

res.end('<h1 style="color:orange; text-align:center;">Good Afternoon!</h1>');

});

// Start server on port 3000

server.listen(3000, () => {

console.log('Server running at http://localhost:3000/');

});

## **▶️ How to Run**

1. Save the file as afternoonServer.js.
2. Open a terminal and run:  
     
    node afternoonServer.js
3. Write a NODE JS Script to demonstrate Chalk NPM

## **✅ Step 1: Initialize Your Project**

**Open a terminal and run:**

**npm init -y**

**This creates a package.json file.**

## **✅ Step 2: Install Chalk**

**npm install chalk**

## **✅ Step 3: Create the Script**

### **📄 chalkDemo.js**

**// chalkDemo.js**

**// Import chalk (v5+ uses ESM, but we'll use CommonJS-compatible syntax for simplicity)**

**import chalk from 'chalk';**

**// Display colored messages**

**console.log(chalk.green('✅ Success! This is a green message.'));**

**console.log(chalk.red('❌ Error! Something went wrong.'));**

**console.log(chalk.blue('ℹ️ Info: This is an informational message.'));**

**console.log(chalk.yellow('⚠️ Warning: Be careful!'));**

**// Combine styles**

**console.log(chalk.bold.underline.magenta('This is bold, underlined, and magenta.'));**

**// Background colors**

**console.log(chalk.black.bgCyan('Text with cyan background.'));**

**// Template literals + chalk**

**const username = 'Alice';**

**console.log(chalk`Hello, {bold.green ${username}}! Welcome to {bgYellow.black Node.js}.`);**

**SLIP 5**

1. Create a NODE JS Server to Read content of the File and Display on Browser

## **✅ Step 1: Prepare a Text File**

Create a file named sample.txt in the same directory with some content, for example:

Hello, this is the content of the file!

Welcome to Node.js file reading demo.

## **✅ Step 2: Create the Server Script**

### **📄 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) => {

// Read the file content asynchronously

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

if (err) {

res.writeHead(500, { 'Content-Type': 'text/plain' });

res.end('Error reading the file.');

return;

}

// Send the file content as the response

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

res.end(data);

});

});

const PORT = 3000;

server.listen(PORT, () => {

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

});

## **▶️ How to Run**

1. Make sure sample.txt and fileServer.js are in the same folder.
2. Run the server:

node fileServer.js

1. Open your browser and visit:

http://localhost:3000/

2) Create a NODE JS Server to Write content in the File and Display result on Browser

## **✅ Step 1: Create the Server Script**

### **📄 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) => {

// The content to write in the file

const content = "Hello! This content was written by Node.js server.\nTimestamp: " + new Date().toString();

// Write content to the file asynchronously

fs.writeFile(filePath, content, 'utf8', (err) => {

if (err) {

res.writeHead(500, { 'Content-Type': 'text/plain' });

res.end('Error writing to file.');

return;

}

// On success, display a message in the browser

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

res.end(`<h1>File written successfully!</h1><p>Content written to ${filePath}</p>`);

});

});

const PORT = 3000;

server.listen(PORT, () => {

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

});

## **▶️ How to Run**

1. Save the file as [writeFileServer.js](http://writefileserver.js).
2. Open a terminal and run:  
   node writeFileServer.js
3. Open your browser and go to:  
   <http://localhost:3000/>