**Slip 1**

**Q1) JavaScript Program to display Timer (timer.html)**

📄 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>

▶️ Run karne ka tarika:

* File ko timer.html naam se save kar.
* Browser me open kar (double click).
* Timer start ho jayega. ✅

**Q2) Simple Node Server Script to find addition of 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 of ${num1} and ${num2} = ${num1 + num2}`);

} else {

res.end("Use this format: http://localhost:3000/?num1=10&num2=20");

}

});

server.listen(3000, () => {

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

});

**Slip 2**

**Q1) JavaScript Program to Validation of User Name and Age**

📄 validate.html

<!DOCTYPE html>

<html>

<head>

<title>User Validation</title>

</head>

<body>

<h1>User Form</h1>

<form onsubmit="return validateForm()">

<label>Username:</label>

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

<label>Age:</label>

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

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

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

</form>

<script>

function validateForm() {

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

const age = parseInt(document.getElementById("age").value.trim());

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

errorDiv.innerHTML = "";

if (username.length < 3) {

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

return false;

}

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

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

return false;

}

alert("Validation successful!\\nUsername: " + username + "\\nAge: " + age);

return true;

}

</script>

</body>

</html>

▶️ Run karne ka tarika:

* File ko validate.html naam se save kar.
* Browser me open kar aur form fill karke test kar. ✅

**Q2) Create a Node.js module (sayHello & fact) & use it**

📄 myModule.js

function sayHello(name) {

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

}

function fact(n) {

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

let result = 1;

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

result \*= i;

}

return result;

}

module.exports = { sayHello, fact };

📄 app.js

const myModule = require("./myModule");

console.log(myModule.sayHello("Alice"));

console.log("Factorial of 5 =", myModule.fact(5));

▶️ Run karne ka tarika:

* Dono file (myModule.js aur app.js) ek hi folder me rakho.
* Terminal me chalao:
* node app.js
* Output:
* Hello, Alice! Welcome to Node.js.

Factorial of 5 = 120

**Q1) Create Node.js user defined module MyDateModule.js & access it**

📄 MyDateModule.js

function getCurrentDate() {

const now = new Date();

return now.toISOString().split("T")[0]; // YYYY-MM-DD

}

function getCurrentTime() {

const now = new Date();

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

}

function getDayName() {

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

return days[new Date().getDay()];

}

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("Today is:", dateModule.getDayName());

▶️ Run karne ka tarika:

* Dono file ek hi folder me rakho.
* Terminal me chalao:
* node app.js
* Output (example):
* Current Date: 2025-09-25
* Current Time: 14:30:45
* Today is: Thursday

**Q2) Create Node.js user defined 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 strUtils = require("./myStringUpperReverseModule");

const input = "hello world";

console.log("Original:", input);

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

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

▶️ Run karne ka tarika:

* Dono file ek hi folder me rakho.
* Terminal me chalao:
* node app.js
* Output:
* Original: hello world
* Uppercase: HELLO WORLD

Reversed: dlrow olleh

**Slip 4**

**Q1) Write a 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/");

});

▶️ Run karne ka tarika:

1. File ko afternoonServer.js naam se save kar.
2. Terminal me chalao:
3. node afternoonServer.js
4. Browser open karo:  
   👉 http://localhost:3000/
5. Output: Screen pe **Good Afternoon!** orange color me show hoga. ✅

**Q2) Write a Node.js Script to demonstrate Chalk NPM**

📄 chalkDemo.js

// Step 1: Run these commands once before using:

// npm init -y

// npm install chalk

const chalk = require("chalk");

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!"));

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

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

const username = "Alice";

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

▶️ Run karne ka tarika:

1. File ko chalkDemo.js naam se save kar.
2. Terminal me commands chalao:
3. npm init -y
4. npm install chalk
5. node chalkDemo.js
6. Output: Terminal me colorful messages print honge. ✅

**Slip 5**

**Q1) Create a 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/");

});

📄 sample.txt

Hello, this is a sample file.

Welcome to Node.js File Reading Demo!

▶️ Run karne ka tarika:

1. Dono file (fileServer.js aur sample.txt) ek hi folder me rakho.
2. Terminal me run karo:
3. node fileServer.js
4. Browser me open karo:  
   👉 http://localhost:3000/
5. Output: sample.txt ka content browser me show hoga. ✅

**Q2) Create a 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! This content was 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>Saved to ${filePath}</p>`);

});

});

server.listen(3000, () => {

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

});

▶️ Run karne ka tarika:

1. File ko writeFileServer.js naam se save kar.
2. Terminal me run karo:
3. node writeFileServer.js
4. Browser me open karo:  
   👉 http://localhost:3000/
5. Output: Browser me message aayega **File written successfully!**
6. Saath hi ek output.txt file create/update ho jayegi. ✅

**Slip 6**

**Q1) Write a 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/");

});

▶️ Run karne ka tarika:

1. File ko afternoonServer.js naam se save kar.
2. Terminal me run karo:
3. node afternoonServer.js
4. Browser open karo:  
   👉 http://localhost:3000/
5. Output: Browser me orange color me **Good Afternoon!** show hoga. ✅

**Q2) Create a 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! This content was 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>Saved to ${filePath}</p>`);

});

});

server.listen(3000, () => {

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

});

▶️ Run karne ka tarika:

1. File ko writeFileServer.js naam se save kar.
2. Terminal me run karo:
3. node writeFileServer.js
4. Browser open karo:  
   👉 http://localhost:3000/
5. Output: Browser me message aayega **File written successfully!**
6. Aur ek output.txt file create/update ho jayegi. ✅

**Slip 7**

**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>

▶️ Run karne ka tarika:

* File ko timer.html naam se save kar.
* Browser me open karo (double click).
* Timer start ho jayega. ✅

**Q2) Create a Node.js Module (sayHello & fact) & use it**

📄 myModule.js

function sayHello(name) {

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

}

function fact(n) {

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

let result = 1;

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

result \*= i;

}

return result;

}

module.exports = { sayHello, fact };

📄 app.js

const myModule = require("./myModule");

console.log(myModule.sayHello("Alice"));

console.log("Factorial of 5 =", myModule.fact(5));

▶️ Run karne ka tarika:

1. myModule.js aur app.js ek hi folder me rakho.
2. Terminal me run karo:
3. node app.js
4. Output:
5. Hello, Alice! Welcome to Node.js.

Factorial of 5 = 120

**Slip 8**

**Q1) Create Node.js user defined module MyDateModule.js & access it**

📄 MyDateModule.js

function getCurrentDate() {

return new Date().toISOString().split("T")[0]; // YYYY-MM-DD

}

function getCurrentTime() {

return new Date().toTimeString().split(" ")[0]; // HH:MM:SS

}

function getDayName() {

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

return days[new Date().getDay()];

}

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("Today is:", dateModule.getDayName());

▶️ Run karne ka tarika:

1. Dono file ek hi folder me rakho.
2. Terminal me run karo:
3. node app.js
4. Output example:
5. Current Date: 2025-09-25
6. Current Time: 15:42:12
7. Today is: Thursday

**Q2) Implement CRUD (Create, Read, Update, Delete) using Node.js and Express**

📄 crudServer.js

// Step 1: Run in terminal once before starting:

// npm init -y

// npm install express body-parser

const express = require("express");

const bodyParser = require("body-parser");

const app = express();

app.use(bodyParser.json());

let items = []; // In-memory data

// Create

app.post("/items", (req, res) => {

const item = { id: items.length + 1, name: req.body.name };

items.push(item);

res.json(item);

});

// Read all

app.get("/items", (req, res) => {

res.json(items);

});

// Read single

app.get("/items/:id", (req, res) => {

const item = items.find(i => i.id == req.params.id);

if (!item) return res.status(404).send("Item not found");

res.json(item);

});

// Update

app.put("/items/:id", (req, res) => {

const item = items.find(i => i.id == req.params.id);

if (!item) return res.status(404).send("Item not found");

item.name = req.body.name;

res.json(item);

});

// Delete

app.delete("/items/:id", (req, res) => {

items = items.filter(i => i.id != req.params.id);

res.send("Item deleted");

});

app.listen(3000, () => {

console.log("✅ CRUD server running at http://localhost:3000/");

});

▶️ Run karne ka tarika:

1. File ko crudServer.js naam se save kar.
2. Terminal me run karo:
3. npm init -y
4. npm install express body-parser
5. node crudServer.js
6. API test karne ke liye **Postman** ya browser use karo:
   * **Create (POST)** → http://localhost:3000/items with JSON { "name": "Book" }
   * **Read (GET)** → http://localhost:3000/items
   * **Update (PUT)** → http://localhost:3000/items/1 with JSON { "name": "Notebook" }
   * **Delete (DELETE)** → http://localhost:3000/items/1

**Slip 9**

**Q1) Write a Node.js Server to wish (Good Morning / Good Afternoon) according to Server Time**

📄 wishServer.js

const http = require("http");

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

const hour = new Date().getHours();

let message = "";

if (hour < 12) {

message = "Good Morning!";

} else if (hour < 18) {

message = "Good Afternoon!";

} else {

message = "Good Evening!";

}

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

res.end(`<h1 style="color:green; text-align:center;">${message}</h1>`);

});

server.listen(3000, () => {

console.log("✅ Wish server running at http://localhost:3000/");

});

▶️ Run karne ka tarika:

1. File ko wishServer.js naam se save kar.
2. Terminal me run karo:
3. node wishServer.js
4. Browser open karo:  
   👉 http://localhost:3000/
5. Output: Server time ke hisaab se **Good Morning / Afternoon / Evening** show hoga. ✅

**Q2) Build a Node.js Server that accepts file uploads from a simple HTML form**

📄 uploadServer.js

// Step 1: Run in terminal once before starting:

// npm init -y

// npm install express multer

const express = require("express");

const multer = require("multer");

const path = require("path");

const app = express();

// Configure multer for file upload

const storage = multer.diskStorage({

destination: function (req, file, cb) {

cb(null, "uploads/");

},

filename: function (req, file, cb) {

cb(null, Date.now() + path.extname(file.originalname));

},

});

const upload = multer({ storage });

// Serve upload form

app.get("/", (req, res) => {

res.send(`

<h1>File Upload Form</h1>

<form action="/upload" method="post" enctype="multipart/form-data">

<input type="file" name="myfile" />

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

</form>

`);

});

// Handle file upload

app.post("/upload", upload.single("myfile"), (req, res) => {

if (!req.file) {

return res.send("No file uploaded!");

}

res.send(`✅ File uploaded successfully! Saved as: ${req.file.filename}`);

});

// Create uploads folder if missing

const fs = require("fs");

if (!fs.existsSync("uploads")) {

fs.mkdirSync("uploads");

}

app.listen(3000, () => {

console.log("✅ Upload server running at http://localhost:3000/");

});

▶️ Run karne ka tarika:

1. File ko uploadServer.js naam se save kar.
2. Terminal me run karo:
3. npm init -y
4. npm install express multer
5. node uploadServer.js
6. Browser me open karo:  
   👉 http://localhost:3000/
7. Ek file select karke **Upload** karo.
8. File uploads/ folder me save ho jayegi. ✅

**Slip 10**

**Q1) Write a JavaScript Program to Validate User Name and Password**

📄 validateUser.html

<!DOCTYPE html>

<html>

<head>

<title>User Validation</title>

</head>

<body>

<h1>Login Form</h1>

<form onsubmit="return validateForm()">

<label>Username:</label>

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

<label>Password:</label>

<input type="password" id="password" required><br><br>

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

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

</form>

<script>

function validateForm() {

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

const password = document.getElementById("password").value.trim();

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

errorDiv.innerHTML = "";

if (username.length < 3) {

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

return false;

}

if (password.length < 6) {

errorDiv.textContent = "Password must be at least 6 characters long.";

return false;

}

alert("✅ Validation successful!\nUsername: " + username);

return true;

}

</script>

</body>

</html>

▶️ Run karne ka tarika:

* File ko validateUser.html naam se save kar.
* Browser me open karo aur form test karo. ✅

**Q2) Create a 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! This content was 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>Saved to ${filePath}</p>`);

});

});

server.listen(3000, () => {

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

});

▶️ Run karne ka tarika:

1. File ko writeFileServer.js naam se save kar.
2. Terminal me run karo:
3. node writeFileServer.js
4. Browser me open karo:  
   👉 http://localhost:3000/
5. Output: Browser me message aayega **File written successfully!**
6. Aur ek output.txt file create/update ho jayegi. ✅