

Day 4 Assignment

Name – Soham Sarkar

College – MCKV INSTITUTE OF ENGINEERING



June 15, 2025

**Assignment Day 4: Node.js | Express | Nodemailer | Fake Data | Path | FS**

**1. Express Routing**

**Q1.** Create a basic Express server with two routes:

* GET / that returns "Welcome to Express!"
* GET /about that returns "About Us Page"

**Server.js Code**

* const server = require('express')
* const app = server()
* app.get('/',(req,res)=>{
* res.send(`Welcome to Express!`)
* })
* app.get('/about',(req,res)=>{
* res.send(`About Us Page`)
* })
* app.listen(3000)

**Console**

npm init -y

npm I express nodemon

node server.js

**For running with nodemon go to package.jso file , then in script key after first predefined value put a coma , and then in next line write like this=> “server”:”nodomon server.js”**

npm run server

**2. Nodemailer Setup**

**Q2.** Create a Node.js script using **Nodemailer** that sends an email with:

* Subject: “Assignment Mail”
* Body: “This is sent from Nodemailer!”
* Use your Gmail SMTP with environment variables for credentials.

**mail.js File Code**

* //Require nodemailer for Mail Service
* const nodemailer = require('nodemailer')
* //GMAIL SMTP with Environment variables
* const transporter = nodemailer.createTransport({
* service:'gmail',
* auth:{
* user:'sohamsarkarofficial000@gmail.com',
* pass:'rhoa ovpf shel kkbm'
* }
* })
* //Declaring Reciepients
* const recipients = [
* 'rupamadhikari117@gmail.com',
* 'srutidhara153@gmail.com'
* ]
* //Create Async Function for sending mails
* async function sendBulkEmails(){
* for(const recv of recipients){
* const mailOption={
* from:'sohamsarkarofficial000@gmail.com',
* to:recv,
* subject:"Greeting From SOHAM",
* text: `Hi ${recv},
* Thank you for contacting me through my portfolio. I’ve received your message and will get back to you as soon as possible.
* If you have shared a project idea, feedback, or any inquiry, I truly appreciate your time and interest.
* You can also connect with me on LinkedIn or GitHub (links on my portfolio).
* Have a great day!
* Best regards,
* Soham Sarkar
* (Full-Stack Web Developer)`
* }
* try{
* const info=await transporter.sendMail(mailOption);
* console.log("Mail Sent to Everyone!")
* }catch(err){
* console.log(err)
* }
* }
* }
* sendBulkEmails()

**Console :**

npm I nodemailer

node mail.js

**3. Fake Data Generation**

**Q3. Using the @faker-js/faker package, generate 10 fake user profiles with the following fields:**

* **Full name**
* **Email**
* **Phone number**
* **Address**
* **Save them in an array and log them to the console.**

**faker.js File Code**

* const {faker} = require('@faker-js/faker');
* const fs =  require('fs')
* const generateFakeUserData =(num)=>{
* const users=[];
* for(let i=0;i<=num-1;i++){
* const user = {
* id:i+1,
* name:faker.name.fullName(),
* email:faker.internet.email(),
* phone:faker.phone.number(),
* adress:{
  + street:faker.address.streetAddress(),
  + city:faker.address.city(),
  + country:faker.address.country()
* },
* create\_at:faker.date.past().toISOString()
* }
* users.push(user)
* }
* return users
* }
* const user\_fakeData= generateFakeUserData(10);
* fs.writeFileSync("store\_fake\_user\_data.json",JSON.stringify(user\_fakeData,null,2))
* console.log("fake json user data is created successfully")

**Console:**

npm i @faker-js/faker

node faker.js

**4. Save Fake Data to File**

**Q4. Use the fs module to save the above fake user array to a file named fake\_users.json.**

* const user\_fakeData= generateFakeUserData(10);
* fs.writeFileSync("fake\_users.json",JSON.stringify(user\_fakeData,null,2))
* console.log("fake json user data is created successfully")

**5. Read and Display File Data**

**Q5. Read the content of fake\_users.json and display each user's name and email on the console.**

* // Read the JSON file asynchronously
* fs.readFile('fake\_users.json', 'utf8', (err, data) => {
* if (err) {
* console.error('Error reading file:', err);
* return;
* }
* try {
* const users = JSON.parse(data);
* users.forEach(user => {
* console.log(`Name: ${user.name}, Email: ${user.email}`);
* });
* } catch (parseError) {
* console.error('Error parsing JSON:', parseError);
* }
* });

**6. Path Module Usage**

**Q6. Create a script that:**

* **Joins directory and file names using the path.join() method**
* **Resolves the absolute path of a file data.txt located in a files folder**

**Folder Structure:**

Path\_module/

├── path\_script.js

└── files/

└── data.txt

**path\_script.js**

* const path = require('path');
* // Join directory and file
* const joinedPath = path.join('files', 'data.txt');
* console.log('Joined Path:', joinedPath);
* // Get absolute path
* const absolutePath = path.resolve('files', 'data.txt');
* console.log('Absolute Path:', absolutePath);

**7. Express POST Route**

**Q7. Create an Express POST /contact route to accept a JSON body with name and message, and print it to the console.**

* const express = require('express');
* const nodemailer = require('nodemailer');
* const app = express();
* const port = 3000;
* // Middleware to parse form data (URL-encoded)
* app.use(express.urlencoded({ extended: true }));
* // GET route to serve HTML contact form
* app.get('/contact', (req, res) => {
* res.send(`
* <h2>Contact Me</h2>
* <form action="/contact" method="POST">
* <label for="name">Name:</label><br>
* <input type="text" id="name" name="name" required><br><br>
* <label for="message">Message:</label><br>
* <textarea id="message" name="message" rows="5" required></textarea><br><br>
* <button type="submit">Send</button>
* </form>
* `);
* });
* // POST /contact route
* app.post('/contact', async (req, res) => {
* const { name, message } = req.body;
* if (!name || !message) {
* return res.status(400).send('Name and message are required');
* }
* console.log(`Contact Form Received:\nName: ${name}\nMessage: ${message}`);
* })
* app.listen(port, () => {
* console.log(`Server is running on http://localhost:${port}`);
* });

**8. Email on POST Request**

**Q8. Modify the /contact POST route: When it receives a request, use Nodemailer to send an email to yourself with the received name and message.**

* app.post('/contact', async (req, res) => {
* const { name, message } = req.body;
* if (!name || !message) {
* return res.status(400).send('Name and message are required');
* }
* console.log(`Contact Form Received:\nName: ${name}\nMessage: ${message}`);
* // Nodemailer setup (credentials hardcoded)
* const transporter = nodemailer.createTransport({
* service: 'gmail',
* auth: {
* user: 'sohamsarkarofficial000@gmail.com',
* pass: 'rhoa ovpf shel kkbm'
* }
* });
* const mailOptions = {
* from: 'sohamsarkarofficial000@gmail.com',
* to: 'sohamsarkarofficial000@gmail.com',
* subject: 'New Contact Message from Portfolio',
* text: `Name: ${name}\nMessage: ${message}`
* };
* try {
* await transporter.sendMail(mailOptions);
* res.send('Email sent successfully!');
* } catch (error) {
* console.error('Email sending failed:', error);
* res.status(500).send('Failed to send email.');
* }
* });

**9. Create and Append File**

**Q9. Using fs module:**

* **Check if a file logs.txt exists. If not, create it.**

**Append a line: "App accessed at <current time>" every time your Express server’s root route (/) is hit.**

* const express = require('express');
* const fs = require('fs');
* const path = require('path');
* const app = express();
* const PORT = 3000;
* const logFilePath = path.join(\_\_dirname, 'logs.txt');
* // Check if logs.txt exists, create if not
* if (!fs.existsSync(logFilePath)) {
* fs.writeFileSync(logFilePath, '', 'utf8');
* console.log('logs.txt created');
* }
* // Root route
* app.get('/', (req, res) => {
* const currentTime = new Date().toLocaleString();
* const logLine = `App accessed at ${currentTime}\n`;
* // Append to file
* fs.appendFile(logFilePath, logLine, (err) => {
* if (err) {
* console.error('Error writing to file:', err);
* } else {
* console.log('Log appended');
* }
* });
* res.send('Welcome to the App. Log recorded.');
* });
* // Start server
* app.listen(PORT, () => {
* console.log(`Server running on http://localhost:${PORT}`);
* });

**10. File Upload Directory Creation**

**Q10. Create a folder named uploads using fs.mkdir if it doesn't exist already, and print "Upload directory ready".**

* const fs = require('fs');
* const path = require('path');
* const uploadDir = path.join(\_\_dirname, 'uploads');
* // Check if uploads folder exists
* if (!fs.existsSync(uploadDir)) {
* fs.mkdirSync(uploadDir);
* console.log(' Upload directory created.');
* } else {
* console.log('Upload directory already exists.');
* }
* console.log('Upload directory ready')