Practical 6

Link to my GitHub repository:

https://github.com/RaviThakur322/Node-JS-Practical-GF2023485999/tree/main

1. Write a program to create the server with the dynamic imports with top level await.
2. Goal: Use stream, readline, pipeline, backpressure.
Tasks: Given data/users.csv (~1M rows), count users per domain from email. Write results to out/domains.json without loading all into memory.
3. Create an EventEmitter-based logger with transports and rotation: Logger emits log events; transports subscribe (console, file). Implement size-based rotation at ~50KB per file.

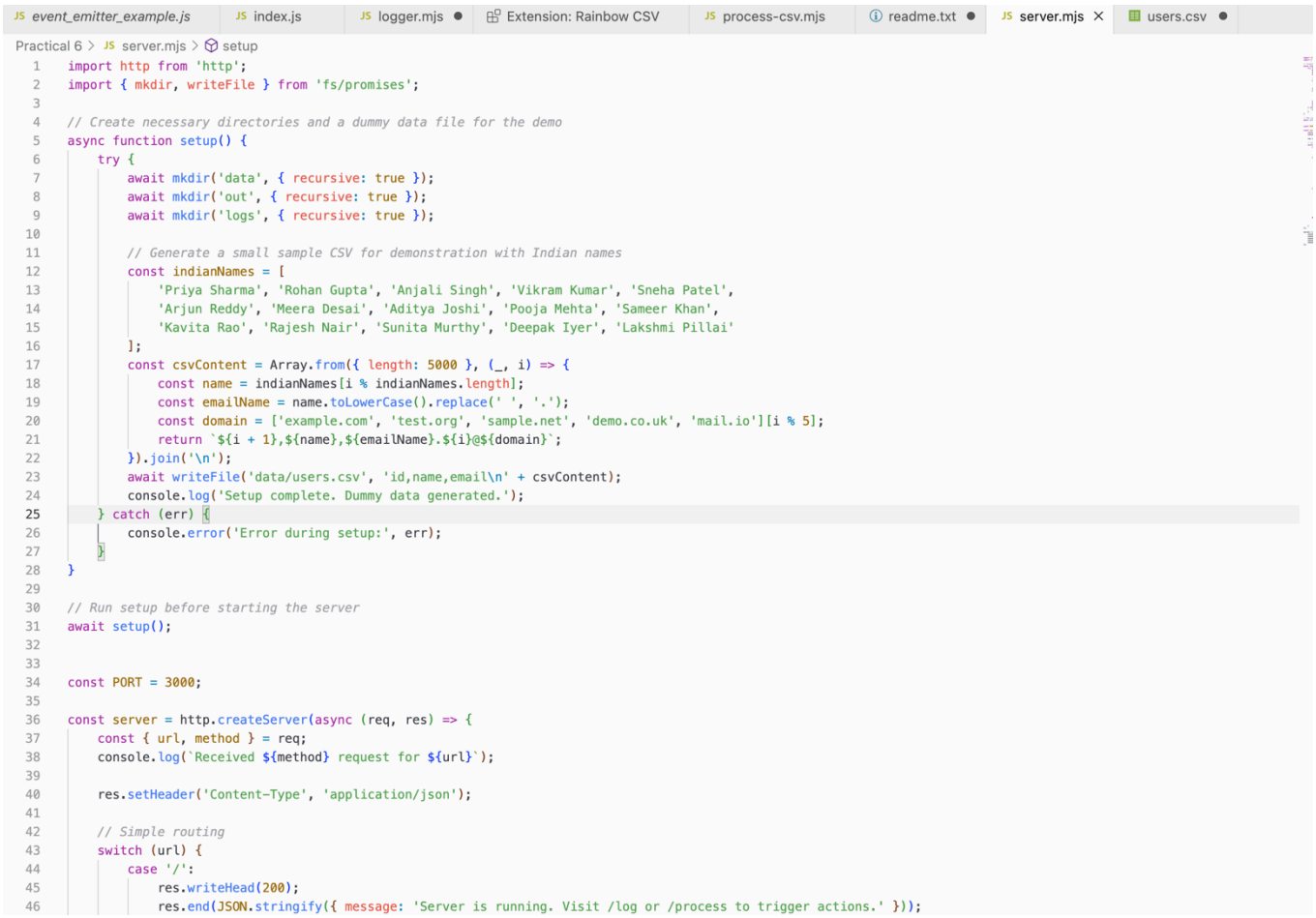
`1.logger.mjs` - An `EventEmitter`-based logger with console and file transports, including size-based log rotation.

```
import http from 'http';
import { mkdir, writeFile } from 'fs/promises';

// Create necessary directories and a dummy data file for the demo
async function setup() {
    try {
        await mkdir('data', { recursive: true });
        await mkdir('out', { recursive: true });
        await mkdir('logs', { recursive: true });

        // Generate a small sample CSV for demonstration with Indian names
        const names = [
            'Priya Sharma', 'Rohan Gupta', 'Anjali Singh', 'Vikram Kumar', 'Sneha Patel',
            'Arjun Reddy', 'Meera Desai', 'Aditya Joshi', 'Pooja Mehta', 'Sameer Khan',
            'Kavita Rao', 'Rajesh Nair', 'Sunita Murthy', 'Deepak Iyer', 'Lakshmi Pillai'
        ];
        const csvContent = Array.from({ length: 5000 }, (_, i) => {
            const name = names[i % names.length];
            const emailName = name.toLowerCase().replace(' ', '.');
            const domain = ['example.com', 'test.org', 'sample.net', 'demo.co.uk', 'mail.io'][i % 5];
            return `${i + 1},${name},${emailName}.${i}@${domain}`;
        }).join('\n');
        await writeFile('data/users.csv', 'id,name,email\n' + csvContent);
        console.log('Setup complete. Dummy data generated.');
    } catch (err) {
        console.error('Error during setup:', err);
    }
}

// Run setup before starting the server
await setup();


const PORT = 3000;

const server = http.createServer(async (req, res) => {
    const { url, method } = req;
    console.log(`Received ${method} request for ${url}`);

    res.setHeader('Content-Type', 'application/json');

    // Simple routing
    switch (url) {
        case '/':
            res.writeHead(200);
            res.end(JSON.stringify({ message: 'Server is running. Visit /log or /process to trigger actions.' }));
```

`2.process-csv.mjs` - A script to efficiently process a large CSV file using streams, handling backpressure.

```
import { mkdir, writeFile } from 'fs/promises';

// Create necessary directories and a dummy data file for the demo
async function setup() {
    try {
        await mkdir('data', { recursive: true });
        await mkdir('out', { recursive: true });
        await mkdir('logs', { recursive: true });

        // Generate a small sample CSV for demonstration with Indian names
        const indianNames = [
            'Priya Sharma', 'Rohan Gupta', 'Anjali Singh', 'Vikram Kumar', 'Sneha Patel',
            'Arjun Reddy', 'Meera Desai', 'Aditya Joshi', 'Pooja Mehta', 'Sameer Khan',
            'Kavita Rao', 'Rajesh Nair', 'Sunita Murthy', 'Deepak Iyer', 'Lakshmi Pillai'
        ];
        const csvContent = Array.from({ length: 5000 }, (_, i) => {
            const name = indianNames[i % indianNames.length];
            const emailName = name.toLowerCase().replace(' ', '.');
            const domain = ['example.com', 'test.org', 'sample.net', 'demo.co.uk', 'mail.io'][i % 5];
            return `${i + 1},${name},${emailName}.${i}@${domain}`;
        }).join('\n');
        await writeFile('data/users.csv', 'id,name,email\n' + csvContent);
        console.log('Setup complete. Dummy data generated.');
    } catch (err) {
        console.error('Error during setup:', err);
    }
}

// Run setup before starting the server
await setup();


const PORT = 3000;

const server = http.createServer(async (req, res) => {
    const { url, method } = req;
    console.log(`Received ${method} request for ${url}`);

    res.setHeader('Content-Type', 'application/json');

    // Simple routing
    switch (url) {
        case '/':
            res.writeHead(200);
            res.end(JSON.stringify({ message: 'Server is running. Visit /log or /process to trigger actions.' }));
```

`3.server.mjs` - An HTTP server that uses dynamic imports and top-level await to run the other modules on demand.



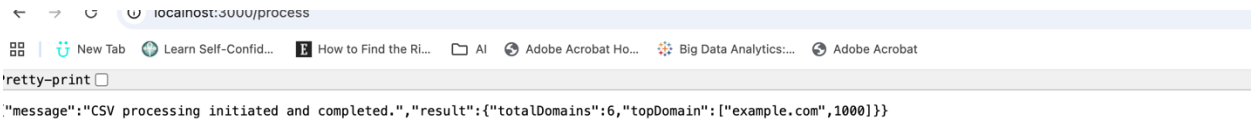
event_emitter_example.js | index.js | logger.mjs | Extension: Rainbow CSV | process-csv.mjs | readme.txt | server.mjs | users.csv

Practical 6 > server.mjs > setup

```
import http from 'http';
import { mkdir, writeFile } from 'fs/promises';

// Create necessary directories and a dummy data file for the demo
async function setup() {
    try {
        await mkdir('data', { recursive: true });
        await mkdir('out', { recursive: true });
        await mkdir('logs', { recursive: true });

        // Generate a small sample CSV for demonstration with Indian names
        const indianNames = [
            'Priya Sharma', 'Rohan Gupta', 'Anjali Singh', 'Vikram Kumar', 'Sneha Patel',
            'Arjun Reddy', 'Meera Desai', 'Aditya Joshi', 'Pooja Mehta', 'Sameer Khan',
            'Kavita Rao', 'Rajesh Nair', 'Sunita Murthy', 'Deepak Iyer', 'Lakshmi Pillai'
        ];
        const csvContent = Array.from({ length: 5000 }, (_, i) => {
            const name = indianNames[i % indianNames.length];
            const emailName = name.toLowerCase().replace(' ', '.');
            const domain = ['example.com', 'test.org', 'sample.net', 'demo.co.uk', 'mail.io'][i % 5];
            return `${i + 1},${name},${emailName}.${i}@${domain}`;
        }).join('\n');
        await writeFile('data/users.csv', 'id,name,email\n' + csvContent);
        console.log('Setup complete. Dummy data generated.');
    } catch (err) {
        console.error('Error during setup:', err);
    }
}

// Run setup before starting the server
await setup();


const PORT = 3000;

const server = http.createServer(async (req, res) => {
    const { url, method } = req;
    console.log(`Received ${method} request for ${url}`);

    res.setHeader('Content-Type', 'application/json');

    // Simple routing
    switch (url) {
        case '/':
            res.writeHead(200);
            res.end(JSON.stringify({ message: 'Server is running. Visit /log or /process to trigger actions.' }));
```

Output:

```
Node.js v24.8.0
ravithakur@Ravis-MacBook-Air Node-JS-Practical-GF2023485999 % node "/Users/ravithakur/Documents/GitHub/Node/Node-JS-Practica
Setup complete. Dummy data generated.
🚀 Server listening on http://localhost:3000
Endpoints:
  - /        : Home
  - /log     : Trigger the logger
  - /process : Trigger the CSV stream processing
Received GET request for /process
Starting CSV processing...
✅ CSV processing complete. Results saved to out/domains.json
```



{"message":"Server is running. Visit /log or /process to trigger actions."}