|  |  |  |
| --- | --- | --- |
| **Name : Harsh Jain** | **Class/Roll No. : D11AD/25** | **Grade :** |

**Title of Experiment**

Node.js ( Asynchronous Programming )

**Objective of Experiment**

To orient students to node for developing front end applications

# Outcome of Experiment

Construct front end applications using React and back end using

Node.js/express.

# Problem Statement

* Write a Node.js program that reads a list of file paths from a text file (one path per line).
* Using asynchronous programming techniques, implement a function to read each file asynchronously and print its content to the console.
* Ensure that the files are read concurrently, not sequentially.
* Handle errors appropriately and log them to a separate error log file.

# Description

Node.js is an open-source, cross-platform JavaScript runtime environment that allows developers to run JavaScript code outside of web browsers. It's built on the V8 JavaScript engine, which is the same engine used by Google Chrome for executing JavaScript. Node.js is designed to be efficient, scalable, and non-blocking, making it well-suited for building server-side applications, networked applications, and real-time applications.

Here are some key concepts and features of Node.js:

* Event-Driven and Non-Blocking I/O: One of the core principles of Node.js is its event-driven, non-blocking architecture. This means that Node.js can handle a large number of concurrent connections without blocking the execution of other code. It uses an event loop to manage asynchronous operations, allowing it to efficiently handle tasks like file I/O, network requests, and database operations without waiting for one task to complete before starting another.
* Single-Threaded: Node.js operates on a single-threaded event loop, which is different from traditional server-side technologies that use multiple threads or processes. This single-threaded model simplifies development but can still take full advantage of multi-core processors through event-driven concurrency.
* NPM (Node Package Manager): Node.js comes with a package manager called NPM, which is one of the largest and most active package ecosystems in the world. It allows developers to easily install, manage, and share reusable libraries and modules for various purposes. NPM simplifies dependency management for Node.js applications.
* Modules: Node.js follows the CommonJS module system, which allows developers to create reusable and encapsulated pieces of code. Modules can be easily imported and used in other parts of the application, promoting modularity and maintainability.
* Built-in Core Modules: Node.js provides a set of built-in core modules for handling common tasks such as file I/O (fs), network operations (http, https, net), and more. These modules make it easy to develop various types of applications without the need for external libraries.
* Community and Ecosystem: Node.js has a vibrant and active community of developers, which has led to the creation of a rich ecosystem of libraries and frameworks. Popular web frameworks like Express.js, Hapi.js, and Nest.js are built on top of Node.js.
* Cross-Platform: Node.js is designed to run on various platforms, including Windows, macOS, and Linux, making it highly portable.
* Real-Time Capabilities: Node.js is well-suited for building real-time applications such as chat applications, online gaming platforms, and streaming services. Libraries like Socket.IO and WebSockets enable real-time communication.
* Scalability: Node.js is known for its scalability, allowing developers to easily scale applications horizontally by adding more servers or instances to handle increased loads.
* Community Support and Learning Resources: Due to its popularity, Node.js has a wealth of learning resources, including documentation, tutorials, online courses, and a supportive online community.

# Code

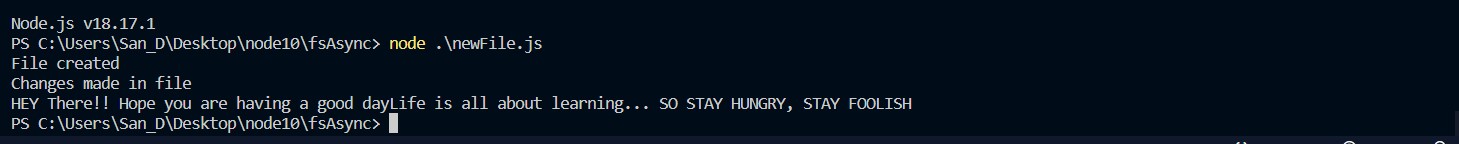
|  |
| --- |
| ***const* fs = require("fs");** |
|  |
| **fs.writeFile('read.txt', "HEY There!! Hope you are having a good day",** |
| **(*err*) *=>* {** |
| **if (*err*) {** |
| **console.error("Error creating file:", *err*);** |
| **} else {**  **console.log("File created");** |
| **}** |
| **});** |
|  |
| **fs.appendFile('read.txt', "Life is all about learning... SO STAY HUNGRY,** |
| **STAY FOOLISH",** |
| **(*err*) *=>* { if (*err*) {** |
| **console.error("Error appending to file:", *err*);** |
| **} else {** |

**Artificial Intelligence and Data Science Department**

**Web Development/Odd Sem 2023-23/Experiment 10**

|  |
| --- |
| **console.log("Changes made in file");** |
| **}** |
| **});** |
|  |
| **fs.readFile('read.txt', "UTF-8",(*err*,*data*) *=>*{** |
| **console.log(*data*);** |
| **});** |

# Output



# Conclusion

Successfully implemented Node.js