PROGRAM – 1

(NODE.JS)

Introduction:

Node.js is an open-source, cross-platform JavaScript runtime environment that allows developers to execute JavaScript code outside of a web browser. Traditionally, JavaScript was confined to client-side web development within browsers. Node.js extends the reach of JavaScript, enabling its use for server-side development, command-line tools, and various other applications.

Key characteristics and features of Node.js include:

- **JavaScript Runtime:**

It utilizes Google Chrome's V8 JavaScript engine, which is known for its high performance in executing JavaScript code.

- **Asynchronous and Event-Driven:**

Node.js operates on an event-driven, non-blocking I/O model. This allows it to handle many concurrent connections efficiently without creating a new thread for each request, making it highly scalable for I/O-bound operations.

- **Single-Threaded Event Loop:**

While Node.js itself is single-threaded, it uses an event loop to manage asynchronous operations, preventing the main thread from blocking while waiting for I/O operations to complete.

- **Versatility:**

Node.js is widely used for building various types of applications, including:

Web servers and APIs, Real-time applications (e.g., chat applications, online gaming), Command-line tools, Microservices, IoT applications

Note: Node.js empowers developers to use JavaScript for full-stack development, leveraging a single language for both client-side and server-side logic.

Node.js Installation procedure:

1. Download the installer:
Go to the official Node.js website: **Node.js**
Download the installer for your operating system (Windows, macOS, or Linux).
Choose either the LTS (Long Term Support) or Current version. LTS is generally recommended for most users.

2. Run the installer:
Locate the downloaded installer file (usually an .msi file for Windows or a .pkg file for macOS).
Double-click the file to start the installation process.
Follow the on-screen instructions, accepting the license agreement and choosing a destination folder if needed.

3. Verify the installation:
Open your terminal or command prompt.
Type node -v and press Enter. This should display the installed Node.js version.
Type npm -v and press Enter. This should display the installed npm (Node Package Manager) version.

a. AIM: Write a program to show the workflow of JavaScript code executable by creating web server in Node.js.

Introduction:

When you view a webpage in your browser, you are making a request to another computer on the internet, which then provides you the webpage as a response. That computer you are talking to via the internet is a *web server*. A web server receives HTTP requests from a client, like your browser, and provides an HTTP response, like an HTML page or JSON from an API.

Step 1: Create folder with any name like project
Step 2: Create new text file and rename with file_name.js like server.js and file open with note pad and paste script below

# CODE

```
const http = require('node:http');


const hostname = '127.0.0.1';

const port = 3000;


const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello, World!\n');
});


server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

Step 4: Open command prompt and type cd path copied like

```
cd C:\Users\fsd\project
```

Step 5: run the server by using node file_name.js like node server.js

It shows like this **Server running at http://127.0.0.1:3000/**

Step 6: Open any browser and paste **http://127.0.0.1:3000/** then it displays

Hello, World!

b. Write a program to transfer data over http protocol using http module.

Key Concepts:

- **HTTP module** is used to create a server that listens for incoming HTTP requests.
- **Data transfer** happens using res.end(data) where data is sent to the client.
- Content-Type is set to application/json for structured data.

Node.js program to **transfer data over the HTTP protocol** using the built-in http module.

#CODE

transferData.js

```
// Load the http module
const http = require('http');

// Define the port for the server
const PORT = 4000;

// Sample data to transfer
const data = {
   id: 101,
   name: "HTTP Data Transfer",
   status: "Success",
   description: "This data is sent over HTTP using Node.js http module"
};

// Create the server
const server = http.createServer((req, res) => {
   console.log(`Request received: ${req.method} ${req.url}`);

   // Set response headers
   res.writeHead(200, {
      'Content-Type': 'application/json',
      'Access-Control-Allow-Origin': '*'
   });
```

```
  // Send the data as JSON
   res.end(JSON.stringify(data));
});

// Start the server
server.listen(PORT, () => {
   console.log(`Server running at http://localhost:${PORT}/`);
});
```

How to Run:

1. Save the code in a file named transferData.js.
2. Open terminal and run: node transferData.js
3. Open your browser and go to:
   http://localhost:4000

OUTPUT:

```
{
  "id": 101,
  "name": "HTTP Data Transfer",
  "status": "Success",
  "description": "This data is sent over HTTP using Node.js http module"
}
```

c. Create a text file src.txt and add the following content to it. (HTML, CSS, Javascript, Typescript, MongoDB, Express.js, React.js, Node.js)

To create a text file named src.txt and add the specified content to it, follow these steps using **Node.js**:

createFile.js

```
const fs = require('fs');
const path = require('path');

// Define the file name and content
const fileName = 'src.txt';
const content = 'HTML, CSS, Javascript, Typescript, MongoDB, Express.js, React.js, Node.js';

// Create the file and write the content
fs.writeFile(path.join(__dirname, fileName), content, (err) => {
  if (err) {
    console.error('Error writing file:', err);
    return;
  }
  console.log(`${fileName} created successfully with content.`);
});
```

Steps to Run:

1. Save the above code in a file named createFile.js.
2. Open your terminal or command prompt.
3. Run the script using: **node createFile.js**
4. **It will create src.txt in the same directory with the following contents:**

📄 src.txt (Output Content)

```
HTML
CSS
JavaScript
TypeScript
MongoDB
Express.js
React.js
Node.js
```