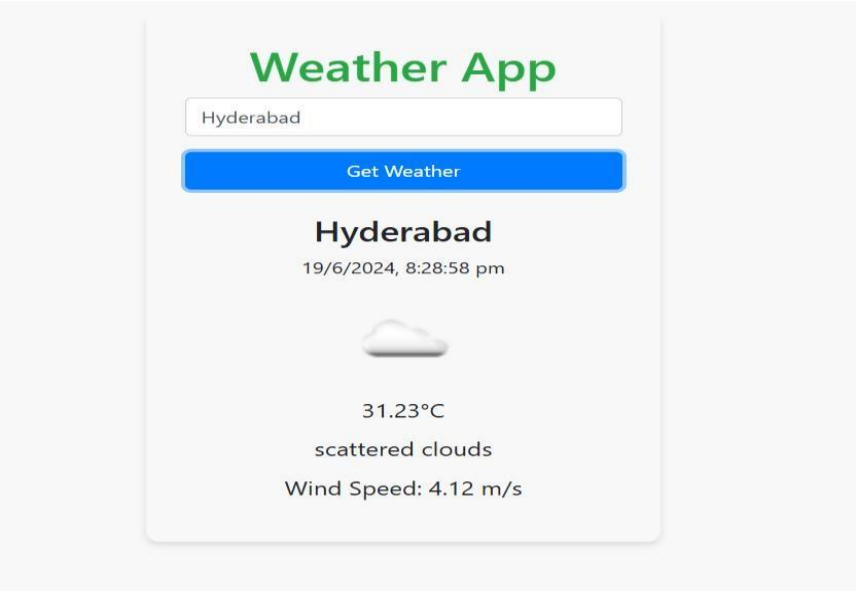
4. Explore the features of ES6 like arrow functions, callbacks, promises, async/await. Implement an application for reading the weather information from openweathermap.org and display the information in the form of a graph on the web page.

```
<html>

<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Weather App</title>
   <link href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css"
rel="stylesheet">        <style>         body {             background-color: #f8f9fa;
        }

        .weather-card {             max-width: 400px;             margin:
50px auto;             padding: 30px;             border-radius: 10px;
box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
        }

        #city-input {             border-radius:
5px;             border-color: #ccc;
        }

        #city-input-btn {             background-
color: #007bff;             border-color: #007bff;
color: #fff;             border-radius: 5px;
        }

        #city-input-btn:hover {
background-color: #0056b3;             border-
color: #0056b3;
        }

        #weather-info {         display:
none;             margin-top: 20px;
        }

        #weather-icon               {
width: 100px;             height:
100px;
        }

        #temperature,
        #description,       #wind-speed {
font-size: 1.2em;             margin-
bottom: 10px;
```

```
}
  </style> </head>

<body>
  <div class="container">
    <div class="weather-card">
      <h1 class="text-success text-center">Weather App</h1>          <input type="text"
id="city-input" class="form-control mb-3" placeholder="Enter city name">
      <button id="city-input-btn" class="btn btn-primary btn-block">Get
Weather</button>
      <div id="weather-info" class="text-center mt-4">
        <h3 id="city-name"></h3>
        <p id="date"></p>
        <img id="weather-icon" src="" alt="Weather Icon">
        <p id="temperature"></p>
        <p id="description"></p>
        <p id="wind-speed"></p>
      </div>
    </div>
  </div>    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></s
cript>

  <script>      const url = 'https://api.openweathermap.org/data/2.5/weather';      const
apiKey = 'dd29a35967661e5b64feb8b437473ec0';

    document.addEventListener('DOMContentLoaded', function () {          const cityInputBtn
= document.getElementById('city-input-btn');          cityInputBtn.addEventListener('click',
function () {            const cityInput = document.getElementById('cityinput').value;
weatherFn(cityInput);
      });
});

    async function weatherFn(cityName) {          const
endpoint =
`${url}?q=${cityName}&appid=${apiKey}&units=metric`;
try {            const response = await fetch(endpoint);
const data = await response.json();            if (response.ok) {
weatherShowFn(data);
        } else {            alert('City not found. Please try again.');
        }
      } catch (error) {          console.error('Error fetching weather data:', error);
      }      }
```

```
function weatherShowFn(data) {            document.getElementById('city-name').textContent =
data.name;            document.getElementById('date').textContent = new
Date().toLocaleString();            document.getElementById('temperature').innerHTML =
`${data.main.temp}°C`;            document.getElementById('description').textContent =
data.weather[0].description;            document.getElementById('wind-speed').innerHTML =
`Wind Speed:
${data.wind.speed} m/s`;            document.getElementById('weather-icon').src =
`http://openweathermap.org/img/w/${data.weather[0].icon}.png`;
document.getElementById('weather-info').style.display = 'block';        }
    </script>
</body>

</html>
```



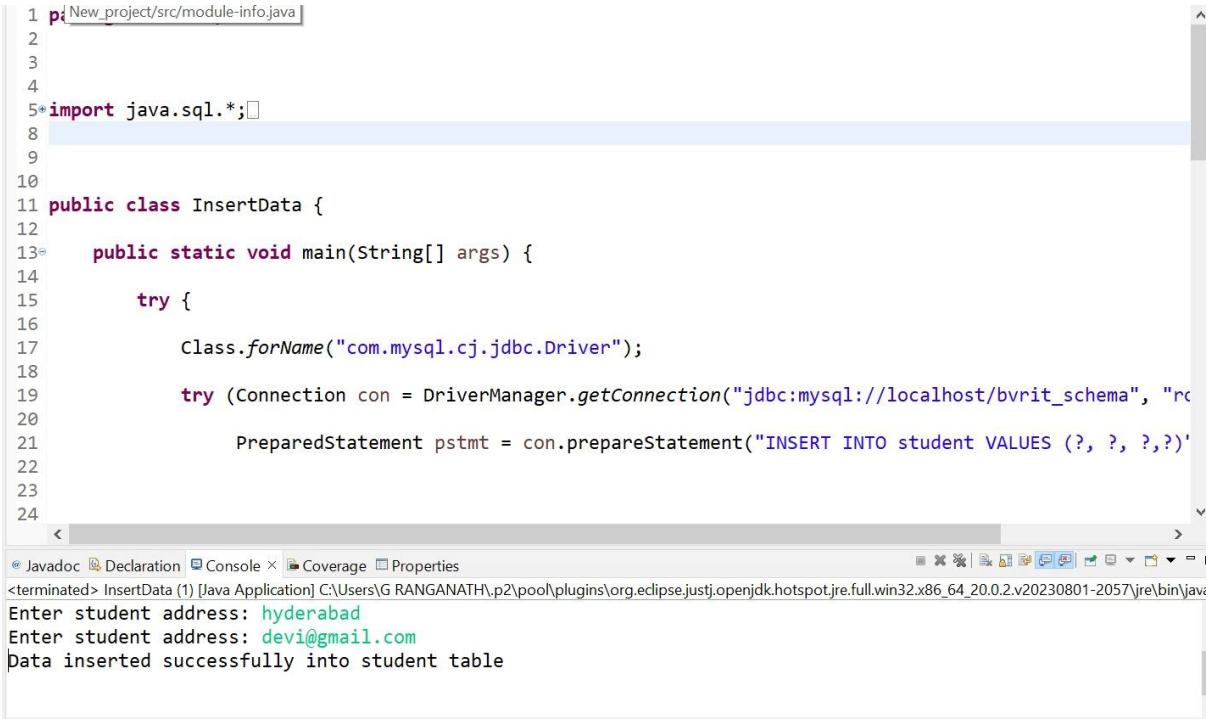
5. Develop a java stand alone application that connects with the database (Oracle / mySql) and perform the CRUD operation on the database tables.

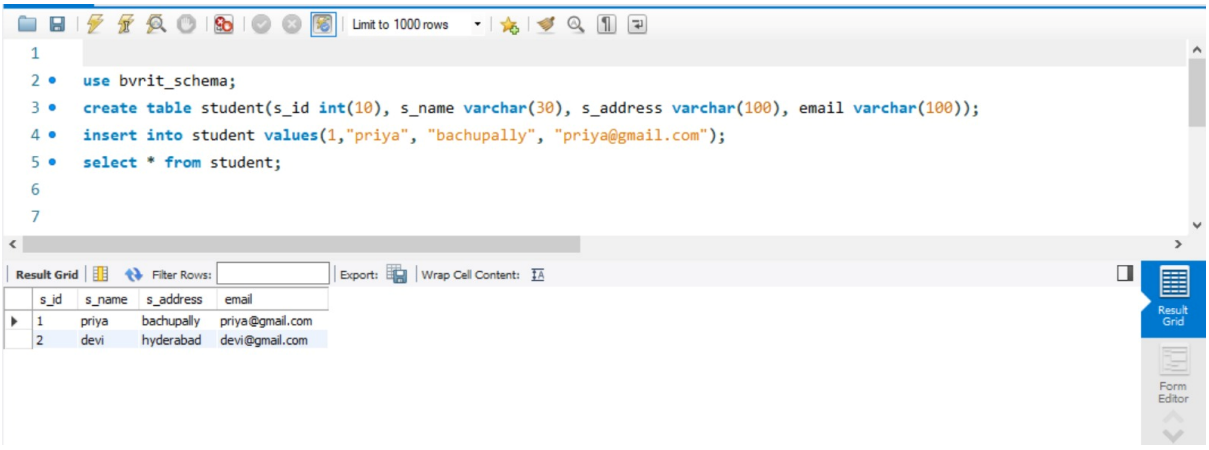
INSERT:

```
import java.sql.*;

import java.util.Scanner;

public class InsertData {

   public static void main(String[] args) {

      try {

         Class.forName("com.mysql.cj.jdbc.Driver");

         try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost/nodejs", "root",
"1234")


PreparedStatement pstmt = con.prepareStatement("INSERT INTO student VALUES (?, ?, ?,?)")) {

               Scanner sc = new Scanner(System.in);

                 System.out.println("Inserting Data into student table:");

               System.out.println("_____");

               System.out.print("Enter student id: ");

               int s_id = sc.nextInt();

               System.out.print("Enter student name: ");


 String s_name = sc.next();
                              System.out.print("Enter student address: ");

               String s_address = sc.next();

               System.out.print("Enter student address: ");

               String email = sc.next();

               pstmt.setInt(1, s_id);

               pstmt.setString(2, s_name);

               pstmt.setString(3, s_address);

               pstmt.setString(4, email);

               pstmt.executeUpdate();
```

```
            System.out.println("Data inserted successfully into
            student table");

        } catch (SQLException err) {

            System.out.println("ERROR: " + err);

        }

    } catch (ClassNotFoundException e) {

        System.out.println("MySQL JDBC driver not found");

    }

  }

}
```

OUTPUT:




Limit to 1000 rows
use bvrit_schema;
create table student(s_id int(10), s_name varchar(30), s_address varchar(100), email varchar(100));
insert into student values(1,"priya", "bachupally", "priya@gmail.com");
select * from student;
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
s_id s_name s_address email
1 priya bachupally priya@gmail.com
2 devi hyderabad devi@gmail.com
Result Grid
Form Editor

DELETE:

```
import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.Scanner;

public class DeleteData {

        public static void main(String[] args) {

    try {

      Class.forName("com.mysql.cj.jdbc.Driver");

      try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost/nodejs", "root",
"1234");

          Statements = con.createStatement()) {

          Scanner sc = new Scanner(System.in);

          System.out.println("Delete Data from student table:");

          System.out.println("_____");

          System.out.print("Enter student id: ");

          int sid = sc.nextInt();

          String deleteQuery = "DELETE FROM student WHERE s_id=" + sid;

          s.executeUpdate(deleteQuery);


         System.out.println("Data deleted successfully");

        } catch (SQLException err) {

          System.out.println("ERROR: " + err);

        }

    } catch (ClassNotFoundException e) {

      System.out.println("MySQL JDBC driver not found");

    }
} }
```

OUTPUTS:

```
package NJSPACK;
import java.sql.Connection;


public class DeleteData {

    public static void main(String[] args) {

        try {

            Class.forName("com.mysql.cj.jdbc.Driver");

            try (Connection con = DriverManager.getConnection("jdbc:mysql://localhost/bvrit_schema", "rc

                    Statement s = con.createStatement()) {



                Scanner sc = new Scanner(System.in);
```

Javadoc | Declaration | Console | Coverage | Properties

<terminated> DeleteData [Java Application] C:\Users\G RANGANATH\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_20.0.2.v20230801-2057\jre\bin\javaw.

```
Enter student id: 1
Data deleted successfully
```

```
use bvrit_schema;
create table student(s_id int(10), s_name varchar(30), s_address varchar(100), email varchar(100));
insert into student values(1,"priya", "bachupally", "priya@gmail.com");
select * from student;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

s_id	s_name	s_address	email
2	devi	hyderabad	devi@gmail.com

6. Create an xml for the bookstore. Validate the same using both DTD and XSD.

bookstore.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE bookstore SYSTEM "bookstore.dtd">
<bookstore>
 <book>
  <title lang="en">Learning XML</title>
  <author>John Doe</author>
  <year>2021</year>
  <price>39.95</price>
 </book>
 <book>
  <title lang="en">Advanced XML</title>
  <author>Jane Smith</author>
  <year>2020</year>
  <price>49.95</price>
 </book>
</bookstore>
```

Create DTD Document

Bookstore.dtd

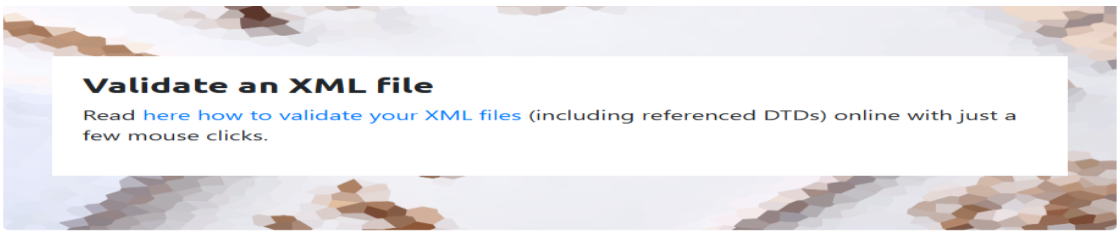
```
<!ELEMENT bookstore (book+)>
<!ELEMENT book (title, author, year, price)>
<!ELEMENT title (#PCDATA)>
<!ATTLIST title lang CDATA #REQUIRED>
<!ELEMENT author (#PCDATA)>
<!ELEMENT year (#PCDATA)>
<!ELEMENT price (#PCDATA)>
```

Create XSD Document bookstore.dtd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="bookstore">
```

```
    <xs:complexType>
      <xs:sequence>
        <xs:element name="book" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="title">
                <xs:complexType>
                  <xs:simpleContent>
                    <xs:extension base="xs:string">
                      <xs:attribute name="lang" type="xs:string" use="required"/>
                    </xs:extension>
                  </xs:simpleContent>
                </xs:complexType>
              </xs:element>
              <xs:element name="author" type="xs:string"/>
              <xs:element name="year" type="xs:integer"/>
              <xs:element name="price" type="xs:decimal"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

OUTPUT:



No errors were found

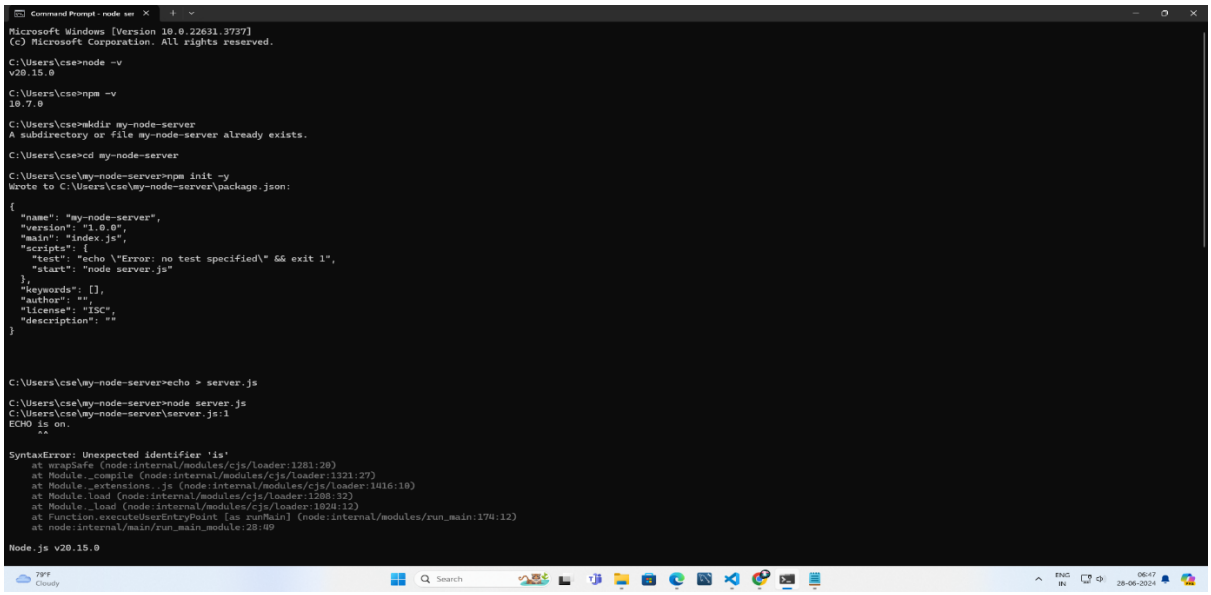
The following files have been uploaded so far:
XML document:
bookstore.dtd

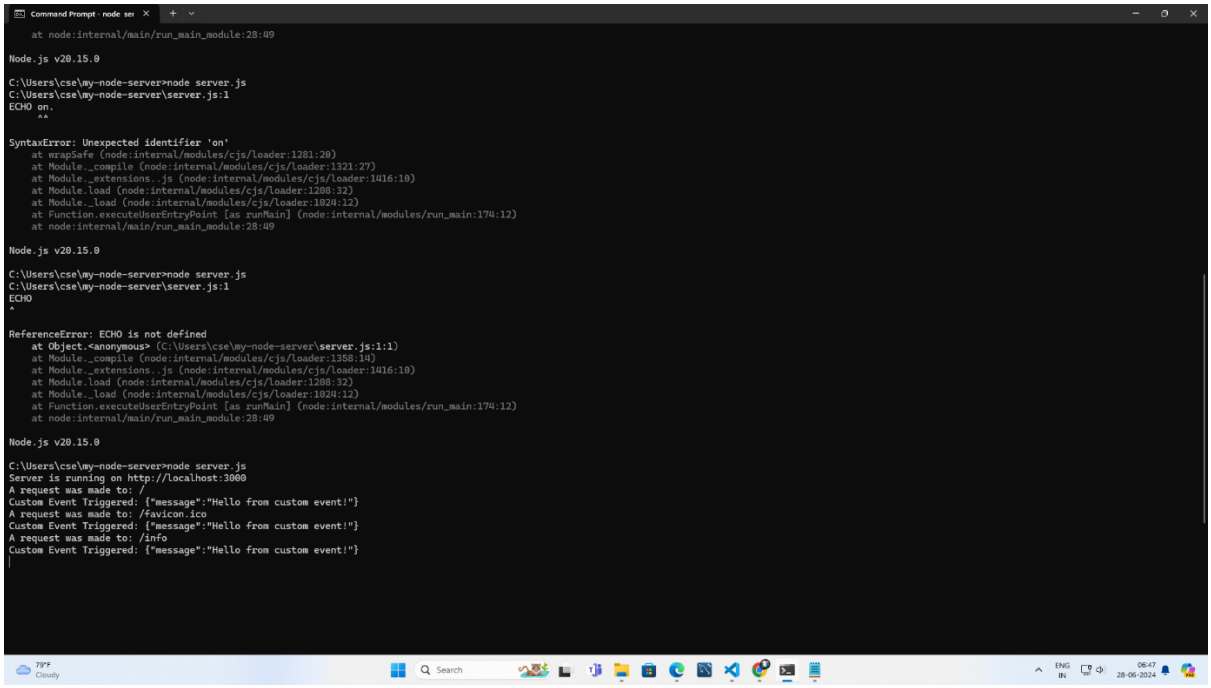
7. Create a custom server using http module and explore the other modules of Node JS like OS, path, event.

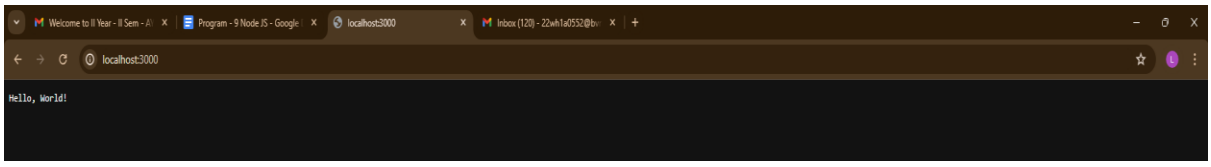
```
// Import the http, os, path, and events modules
const http = require('http');
const os = require('os');
const path = require('path');
const EventEmitter = require('events');
// Create a custom event emitter
class MyEmitter extends EventEmitter {}
const myEmitter = new MyEmitter();
// Define a simple request handler
const requestHandler = (req, res) => {
if (req.url === '/') {
res.writeHead(200, { 'Content-Type': 'text/plain' });
res.end('Hello, World!\n');
} else if (req.url === '/info') {
const info = {
hostname: os.hostname(),
platform: os.platform(),
homedir: os.homedir(),
file: __filename,
dir: __dirname
};
res.writeHead(200, { 'Content-Type': 'application/json' });
res.end(JSON.stringify(info, null, 2));
} else {
res.writeHead(404, { 'Content-Type': 'text/plain' });
res.end('Not Found\n');
```

```
}

// Emit a custom event
myEmitter.emit('requestHandled', req.url);
// Emit the custom message event
myEmitter.emit('customEvent', { message: 'Hello from custom event!' });
};
// Create the server
const server = http.createServer(requestHandler);
// Listen on port 3000
server.listen(3000, () => {
console.log('Server is running on http://localhost:3000');
});
// Listen for the requestHandled event
myEmitter.on('requestHandled', (url) => {
console.log(`A request was made to: ${url}`);
});
// Listen for the customEvent event and log the message
myEmitter.on('customEvent', (data) => {
console.log(`Custom Event Triggered: ${JSON.stringify(data)}`);
});
```

OUTPUT:


Command Prompt - node ser
Microsoft Windows [Version 10.0.22631.3737]
(c) Microsoft Corporation. All rights reserved.
C:\Users\cse>node -v
v20.15.0
C:\Users\cse>npm -v
10.7.0
C:\Users\cse>mkdir my-node-server
A subdirectory or file my-node-server already exists.
C:\Users\cse>cd my-node-server
C:\Users\cse\my-node-server>npm init -y
Wrote to C:\Users\cse\my-node-server\package.json:
{
"name": "my-node-server",
"version": "1.0.0",
"main": "index.js",
"scripts": {
"test": "echo \"Error: no test specified\" && exit 1",
"start": "node server.js"
},
"keywords": [],
"author": "",
"license": "ISC",
"description": ""
}
C:\Users\cse\my-node-server>echo > server.js
C:\Users\cse\my-node-server>node server.js
C:\Users\cse\my-node-server\server.js:1
ECHO is on.
^^
SyntaxError: Unexpected identifier 'is'
at wrapSafe (node:internal/modules/cjs/loader:1281:20)
at Module._compile (node:internal/modules/cjs/loader:1321:27)
at Module._extensions..js (node:internal/modules/cjs/loader:1416:10)
at Module.load (node:internal/modules/cjs/loader:1208:32)
at Module._load (node:internal/modules/cjs/loader:1024:12)
at Function.executeUserEntryPoint [as runMain] (node:internal/modules/run_main:174:12)
at node:internal/main/run_main_module:28:49
Node.js v20.15.0


Command Prompt - node ser
at node:internal/main/run_main_module:28:49
Node.js v20.15.0
C:\Users\cse\my-node-server>node server.js
C:\Users\cse\my-node-server\server.js:1
ECHO on.
^^
SyntaxError: Unexpected identifier 'on'
at wrapSafe (node:internal/modules/cjs/loader:1281:20)
at Module._compile (node:internal/modules/cjs/loader:1321:27)
at Module._extensions..js (node:internal/modules/cjs/loader:1416:10)
at Module.load (node:internal/modules/cjs/loader:1208:32)
at Module._load (node:internal/modules/cjs/loader:1024:12)
at Function.executeUserEntryPoint [as runMain] (node:internal/modules/run_main:174:12)
at node:internal/main/run_main_module:28:49
Node.js v20.15.0
C:\Users\cse\my-node-server>node server.js
C:\Users\cse\my-node-server\server.js:1
ECHO
^
ReferenceError: ECHO is not defined
at Object.<anonymous> (C:\Users\cse\my-node-server\server.js:1:1)
at Module._compile (node:internal/modules/cjs/loader:1358:14)
at Module._extensions..js (node:internal/modules/cjs/loader:1416:10)
at Module.load (node:internal/modules/cjs/loader:1208:32)
at Module._load (node:internal/modules/cjs/loader:1024:12)
at Function.executeUserEntryPoint [as runMain] (node:internal/modules/run_main:174:12)
at node:internal/main/run_main_module:28:49
Node.js v20.15.0
C:\Users\cse\my-node-server>node server.js
Server is running on http://localhost:3000
A request was made to: /
Custom Event Triggered: {"message":"Hello from custom event!"}
A request was made to: /favicon.ico
Custom Event Triggered: {"message":"Hello from custom event!"}
A request was made to: /info
Custom Event Triggered: {"message":"Hello from custom event!"}


localhost:3000
Hello, World!