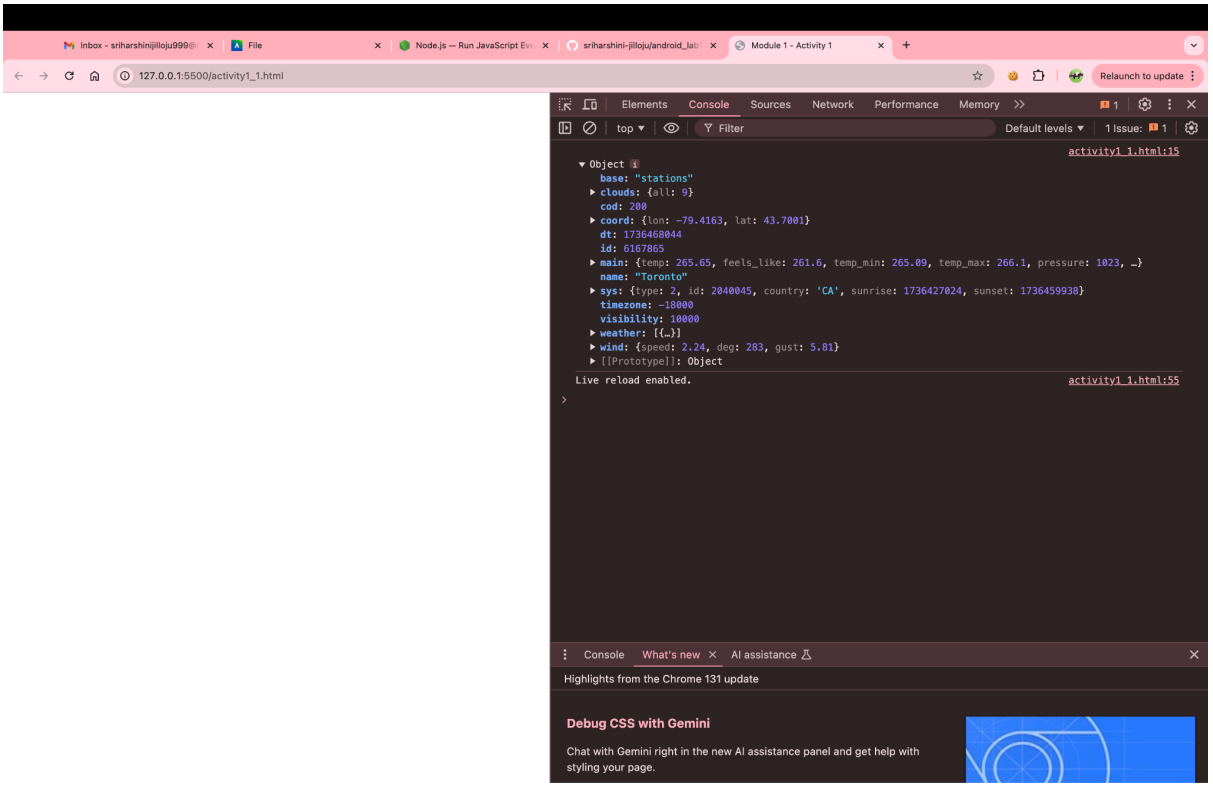
Web Programming & Framework1

Lab-1

Sri Harshini Jilloju - N01649103

Activity-1.1

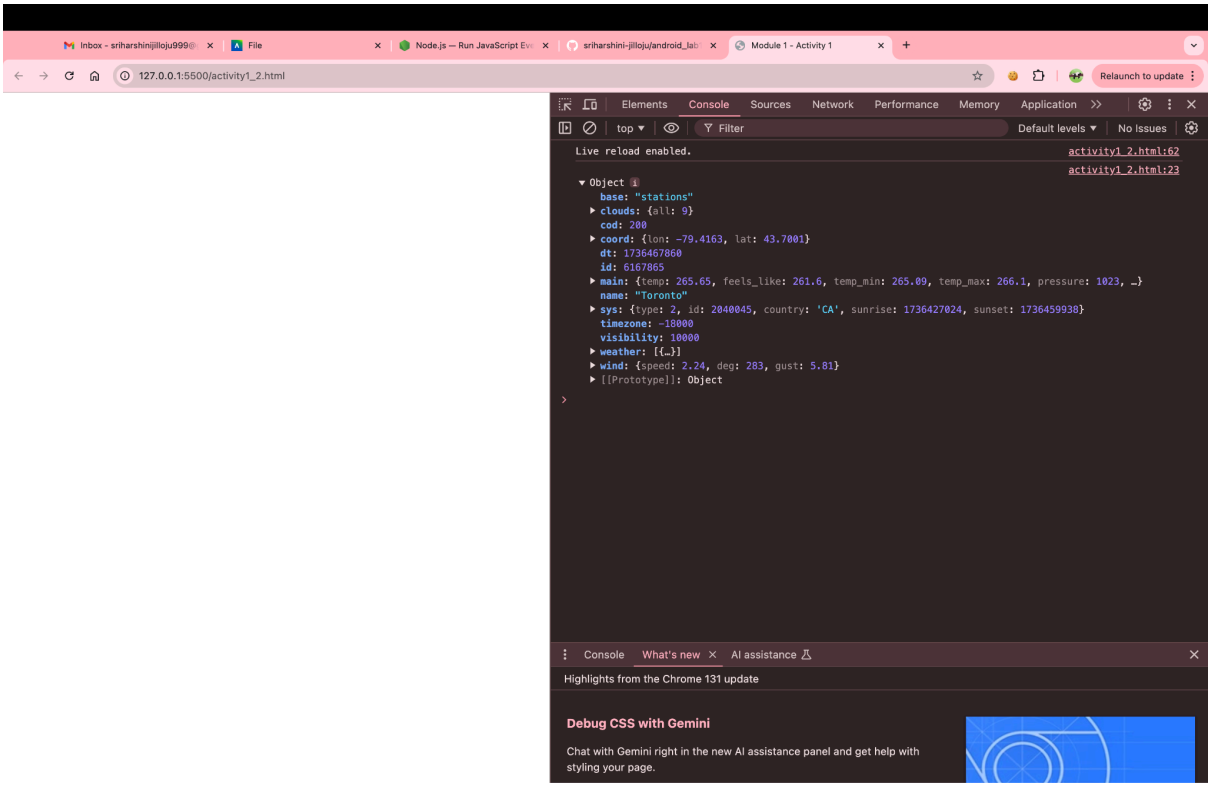
This HTML document fetches weather data for Toronto from the OpenWeatherMap API. Inside the <script> tag, the loadData function is defined to make an HTTP request using the XMLHttpRequest object (xmlhttp). The onreadystatechange event listener monitors the request's progress and ensures the response is processed only when the request is complete (readyState == 4) and successful (status == 200). Once the response is received, it is converted into a JavaScript object using JSON.parse, and the weather data is logged to the console. The xmlhttp.open method sets up a GET request to the API endpoint, including the city name and an API key. The false parameter makes the request synchronous ,and setting it to true makes the request asynchronous. The xmlhttp.send() method sends the request, and the loadData function is immediately called to execute the process.



Activity-1.2

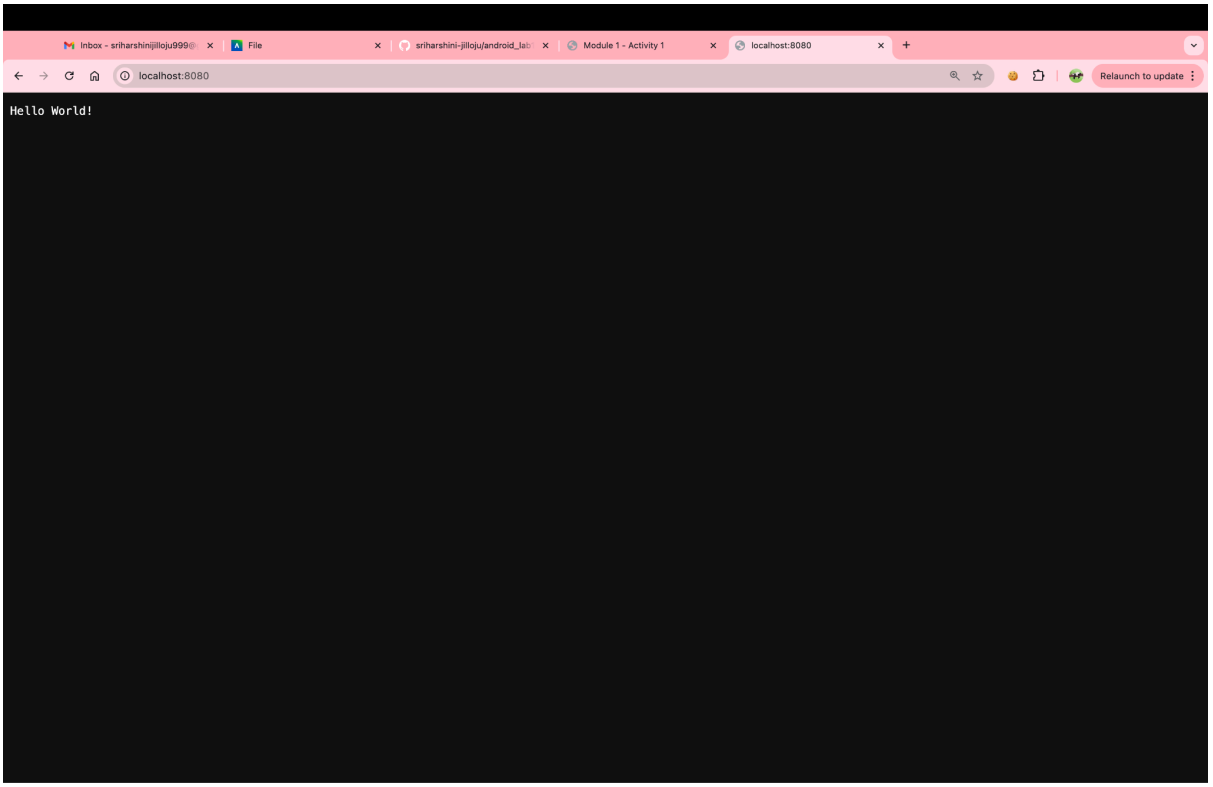
This HTML document retrieves weather data for Toronto from the OpenWeatherMap API using the Fetch API. The loadData function, defined within the <script> tag, makes an asynchronous GET request to the API endpoint. The fetch method returns a Promise, and the first .then block checks if the response is successful (response.ok). If not, it throws an error with the response's status text. If successful, the response is converted to a JSON object

using response.json(). The second .then block processes and logs the weather data to the console. A .catch block handles any errors that occur during the fetch operation by logging an error message. The loadData function is called immediately to fetch the data when the page loads.



Observation:

This code sets up a simple HTTP server in Node.js that listens on port 8080. When you open http://localhost:8080 in your browser, the server sends a response displaying "Hello World!" on the page. It utilizes Node.js's built-in HTTP module to process incoming requests and provide a basic HTML response.


Inbox - sriharshinijilloju999@
File
sriharshini-jilloju/android_lab
Module 1 - Activity 1
localhost:8080
localhost:8080
Relaunch to update
Hello World!

Activity - 3:

Welcome | activity1_1.html | activity1_2.html | myfirst.js | server.js

server.js > ...

```
1  const http = require('http');
2  const hostname = '127.0.0.1';
3  const port = 3000;
4  const server = http.createServer((req, res) => {
5  res.statusCode = 200;
6  res.setHeader('Content-Type', 'text/plain');
7  res.end('Hello World\n');
8  });
9  server.listen(port, hostname, () => {
10 console.log(`Server running at http://${hostname}:${port}/`);
11 });
```

PROBLEMS | OUTPUT | DEBUG CONSOLE | TERMINAL | PORTS

Code

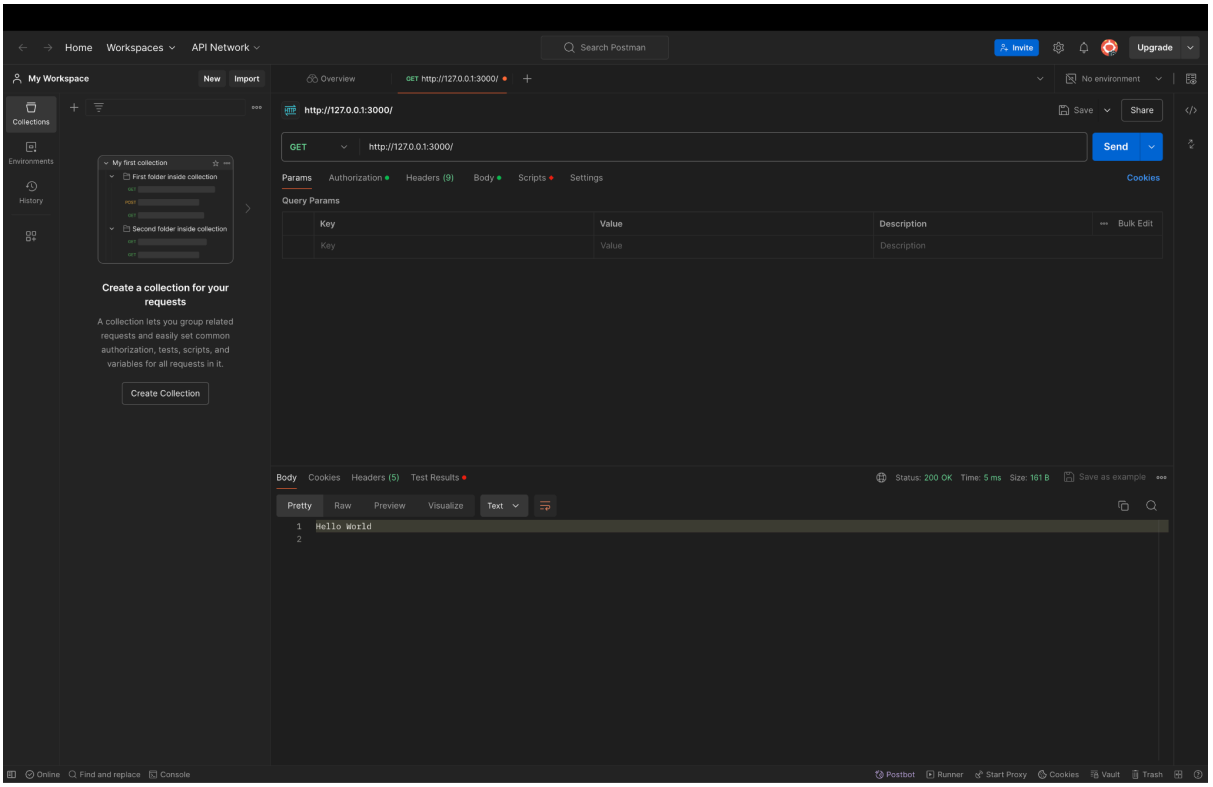
```
[Running] node "/Users/sriharshinijilloju/Desktop/webprogramming_framework1/week1/server.js"
Server running at http://127.0.0.1:3000/

[Done] exited with code=null in 4883.927 seconds

[Running] node "/Users/sriharshinijilloju/Desktop/webprogramming_framework1/week1/myfirst.js"

[Done] exited with code=null in 149.147 seconds

[Running] node "/Users/sriharshinijilloju/Desktop/webprogramming_framework1/week1/server.js"
Server running at http://127.0.0.1:3000/
```



Explanation:
In the provided code, an arrow function is used as the callback for http.createServer, defined as (req, res) => { res.statusCode = 200; res.setHeader('Content-Type', 'text/plain'); res.end('Hello World\n'); }. Arrow functions provide a syntax for defining functions in JavaScript.

string interpolation is utilized in console.log(\Server running at http://${hostname}:${port}/`);,
This uses string interpolation to embed hostname and port variables into the string. backticks () and ${} are used to embed the hostname and port variables directly into the string.