Title : 1. Basic JavaScript Program: Write a simple JavaScript program to perform arithmetic operations.

// Basic JavaScript Program to perform arithmetic operations

let num1 = Number(prompt("Enter the first number:"));

let num2 = Number(prompt("Enter the second number:"));

let addition = num1 + num2;

let subtraction = num1 - num2;

let multiplication = num1 \* num2;

let modulus = num1 % num2;

let division = num2 !== 0 ? (num1 / num2) : "Cannot divide by zero";

console.log("Addition: " + addition);

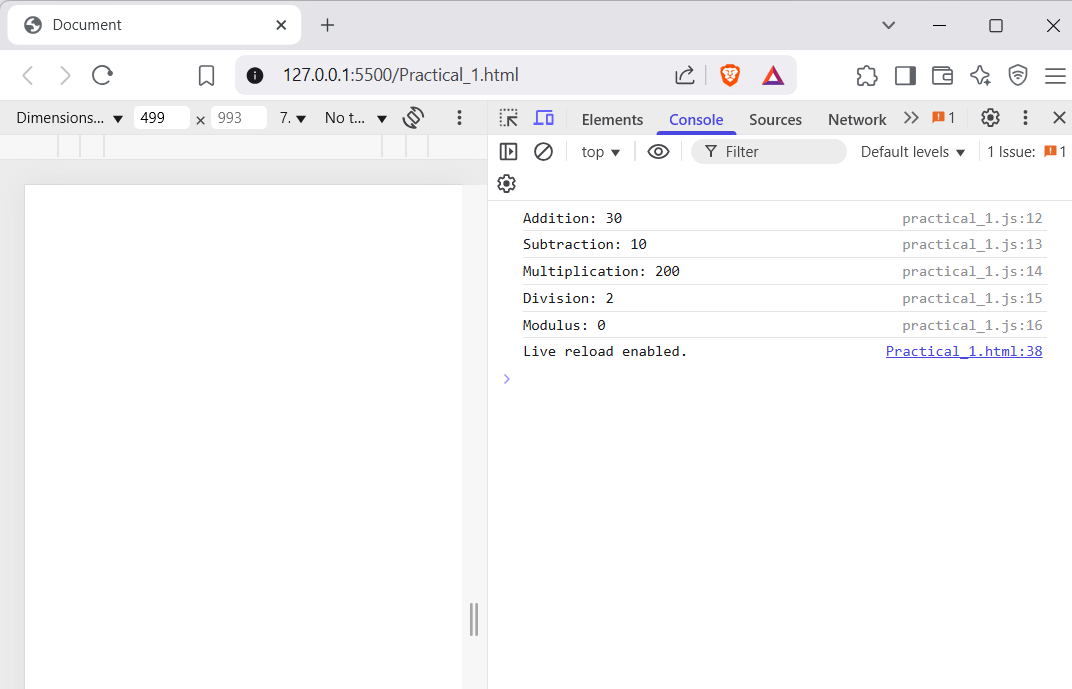
console.log("Subtraction: " + subtraction);

console.log("Multiplication: " + multiplication);

console.log("Division: " + division);

console.log("Modulus: " + modulus);

Output :



Title : 2. Control Structures: Implement a JavaScript program using if-else statements and loops

let number = 5;  // You can change this number to test

if (number > 0) {

    console.log(number + " is a positive number.");

} else if (number < 0) {

    console.log(number + " is a negative number.");

} else {

    console.log("The number is zero.");

}

// Using a for loop to print numbers from 1 to 10

console.log("\nNumbers from 1 to 10:");

for (let i = 1; i <= 10; i++) {

    console.log(i);

}

// Using a while loop to print numbers from 1 to 5

console.log("\nNumbers from 1 to 5 using while loop:");

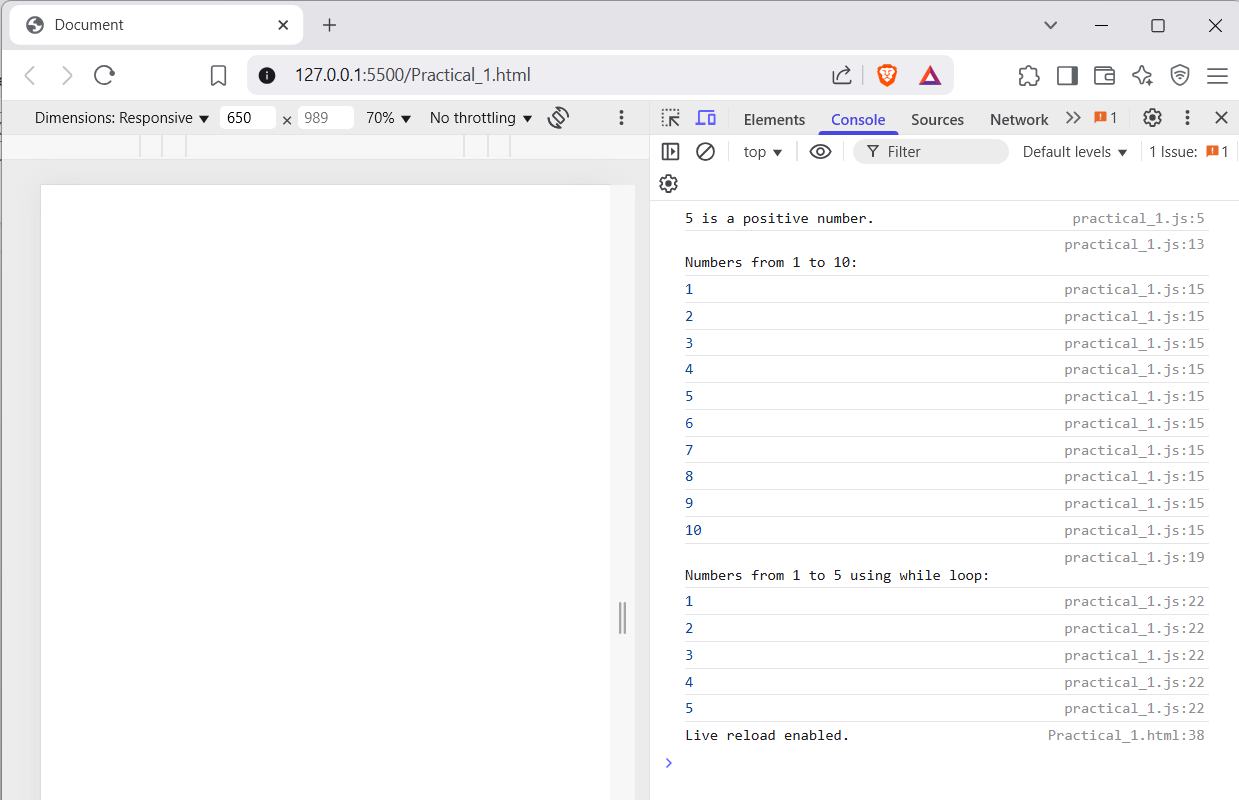
let i = 1;

while (i <= 5) {

    console.log(i);

    i++;

} **Output :**



Title : 3. JavaScript Functions: Create and invoke functions that perform specific tasks

// 1. Named Function (Function Declaration)

function greet() {

    console.log("Hello from a Named Function!");

  }

  greet();

  // 2. Anonymous Function (Function Expression)

  const sayHello = function () {

    console.log("Hello from an Anonymous Function!");

  };

  sayHello();

  // 3. Arrow Function

  const multiply = (a, b) => a \* b;

  console.log("Arrow Function Result (4 \* 5):", multiply(4, 5));

  // 4. Immediately Invoked Function Expression (IIFE)

  (function () {

    console.log("This message is from an IIFE (runs immediately)!");

  })();

  // 5. Function with Parameters and Return

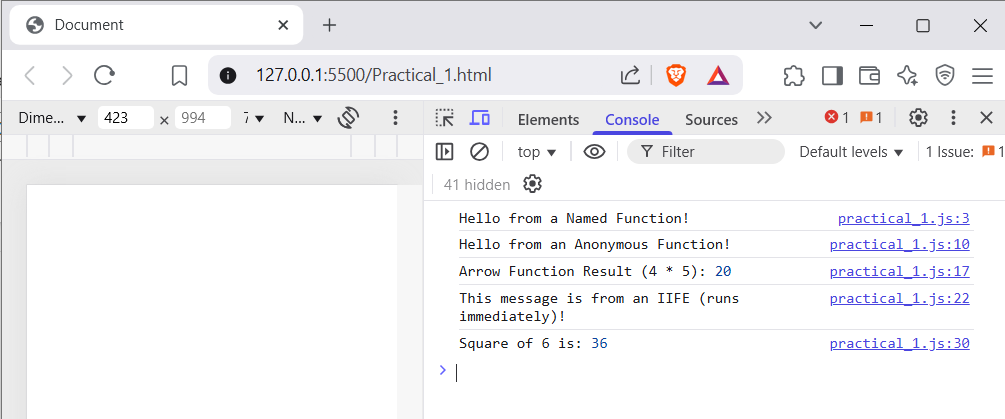
  function square(num) {

    return num \* num;

  }

  console.log("Square of 6 is:", square(6));

Output :



Title : 4. Event Handling: Develop a web page where JavaScript responds to user events (e.g., button clicks)

<!DOCTYPE html>

<html>

<head>

  <title>Highlight Text Example</title>

  <style>

    .highlight {

      background-color: rgba(255, 255, 86, 0.666);

      font-weight: bold;

      padding: 5px;

      border-radius: 4px;

    }

  </style>

</head>

<body>

  <h1>Highlight Text Example</h1>

  <!-- Button to trigger the highlight -->

  <button id="highlightTextButton">Highlight Text</button>

  <!-- Text to be highlighted -->

  <p id="message">This is the original text. Click the button to highlight this text!</p>

  <script>

    // Event handler to highlight text when "Highlight Text" button is clicked

    document.getElementById("highlightTextButton").addEventListener("click", function() {

      const messageElement = document.getElementById("message");

      messageElement.classList.toggle("highlight"); // Toggle highlight class

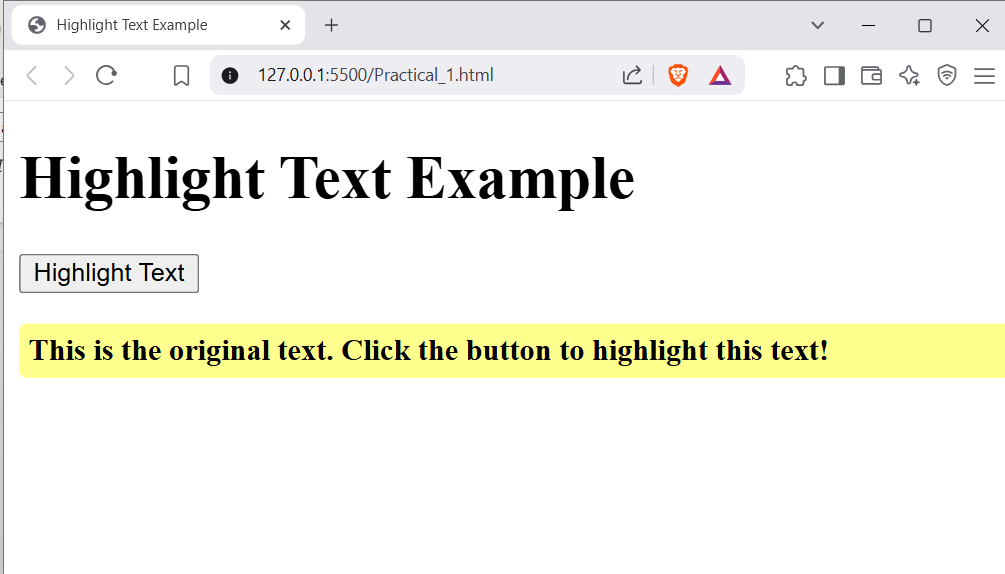
    });

  </script>

</body>

</html>

Output :



Title : 6. Set Up a Basic Node.js Server • Create a basic HTTP server using Node.js that responds with "Hello, World!" to any request

// Import the http module to create an HTTP server

const http = require('http');

// Define the port and host

const port = 3000;

const host = '127.0.0.1';  // Localhost address

// Create an HTTP server that responds with "Hello, World!"

const server = http.createServer((req, res) => {

  res.statusCode = 200;  // HTTP status code 200 (OK)

  res.setHeader('Content-Type', 'text/plain');  // Content type is plain text

  res.end('Hello, World!');  // Send the "Hello, World!" response

});

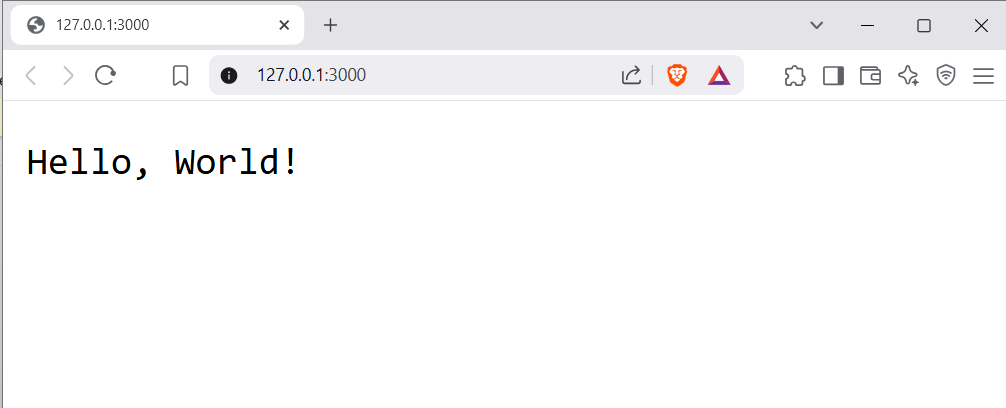
// Start the server and listen on the specified port

server.listen(port, host, () => {

  console.log(`Server is running at http://${host}:${port}/`);

});

Output :



Title : 7. Create a REST API with Node.js • Develop a basic REST API using Node.js that performs CR (Create, Read) operations on a simple data set.

// Import express to create the server

const express = require('express');

// Create an express application

const app = express();

// Define a port number

const port = 3000;

// Use middleware to parse JSON data from requests

app.use(express.json());

// Sample data for demonstration

let users = [

  { id: 1, name: 'Alice', age: 30 },

  { id: 2, name: 'Bob', age: 25 }

];

// 1. Create (POST) operation to add a new user

app.post('/users', (req, res) => {

  const { name, age } = req.body;  // Extract name and age from the request body

  // Generate a new user ID

  const newUser = { id: users.length + 1, name, age };

  // Add the new user to the "users" data

  users.push(newUser);

  // Respond with the newly created user data

  res.status(201).json(newUser);

});

// 2. Read (GET) operation to get all users

app.get('/users', (req, res) => {

  // Respond with all users in the data

  res.status(200).json(users);

});

// 3. Read (GET) operation to get a single user by ID

app.get('/users/:id', (req, res) => {

  const userId = parseInt(req.params.id, 10);

  // Find the user by ID

  const user = users.find(u => u.id === userId);

  if (user) {

    // Respond with the user data

    res.status(200).json(user);

  } else {

    // If the user is not found, return a 404 error

    res.status(404).json({ message: 'User not found' });

  }

});

// Start the server and listen on the specified port

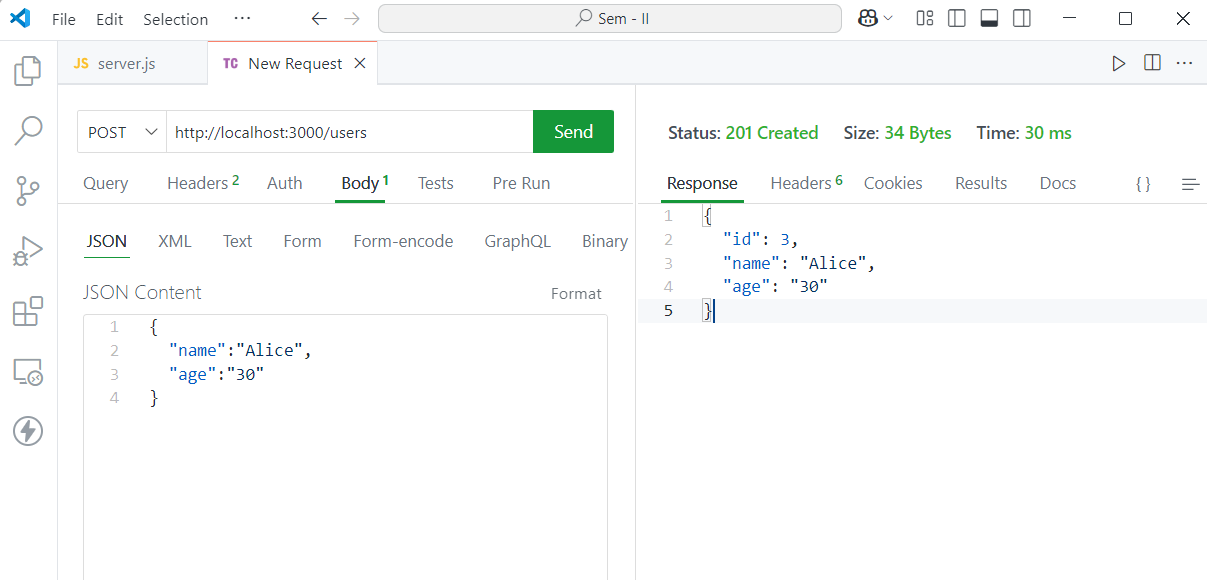
app.listen(port, () => {

  console.log(`Server is running on http://localhost:${port}`);

});

Output :

1 . Post Method :



2. Post Method :

