JS
JavaScript

Class # 1

Introduction

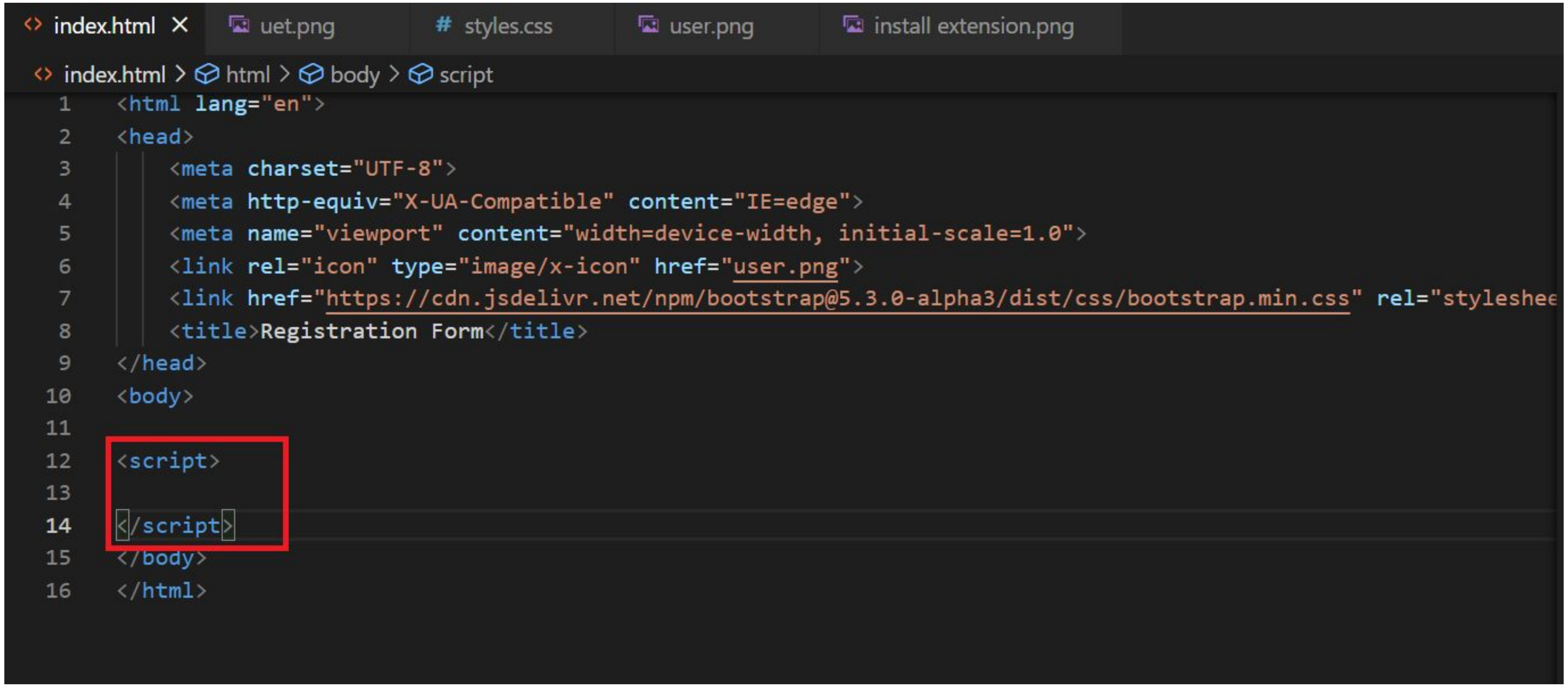
JavaScript is a high-level, interpreted programming language primarily used for web development. It was originally created to add interactivity and dynamic functionality to websites. JavaScript allows developers to manipulate and modify web page content, handle user interactions, and create interactive web applications.

Where to Add Javascript

There are 2 ways to add javascript in HTML

- In HTML File
- Make .js file and import in HTML

Add in HTML File

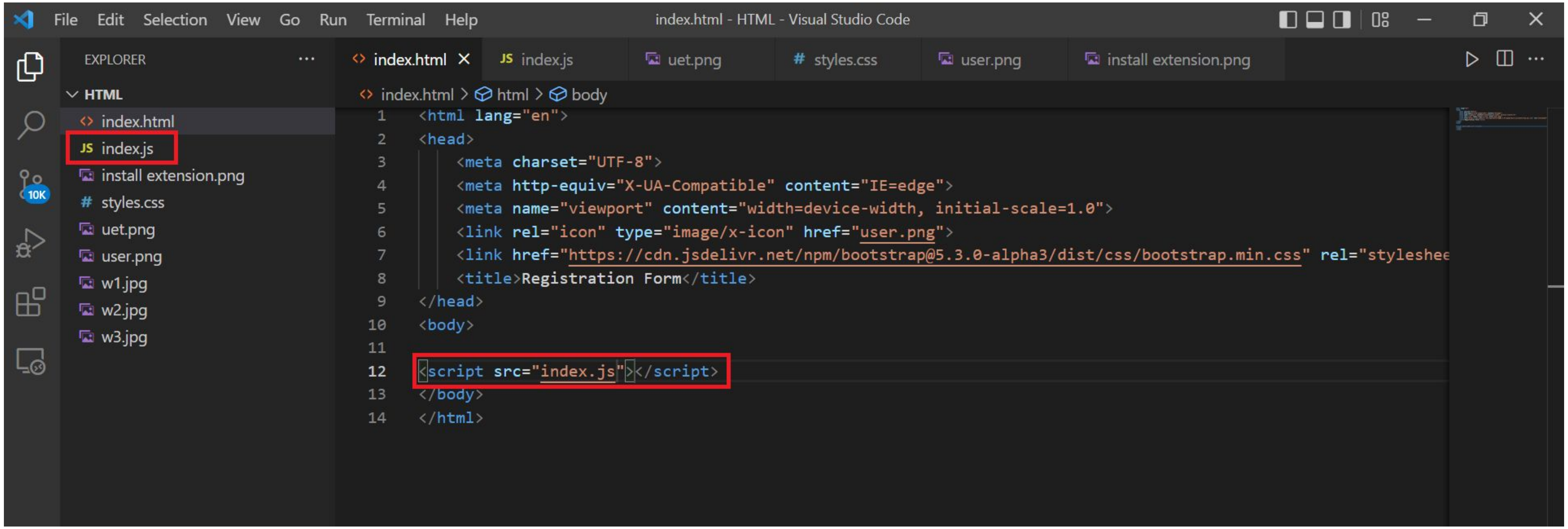


```html
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="icon" type="image/x-icon" href="user.png">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css" rel="styleshee
    <title>Registration Form</title>
</head>
<body>

<script>

</script>
</body>
</html>
```

Import from .js file



```html
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="icon" type="image/x-icon" href="user.png">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css" rel="styleshee
    <title>Registration Form</title>
</head>
<body>

<script src="index.js"></script>
</body>
</html>
```

Where to display JS output?

JavaScript can "display" data in different ways:

- Writing into an HTML element, using innerHTML
- Writing into the HTML output using document.write()
- Writing into an alert box, using window.alert()
- Writing into the browser console, using console.log()

How to get HTML element in JS?

There are different ways to get HTML element in Jacavscript like

document.getElementById(): This method allows you to retrieve an element using its unique id attribute. It returns a reference to the first element with the specified ID.

document.getElementsByClassName(): This method returns a collection of elements that have a specific class name. It returns an HTMLCollection, which is an array-like object.

document.getElementsByTagName(): This method returns a collection of elements with the specified tag name. It retrieves all elements of a particular HTML tag.

How to get HTML element in JS?

There are different ways to get HTML element in Jacavscript like

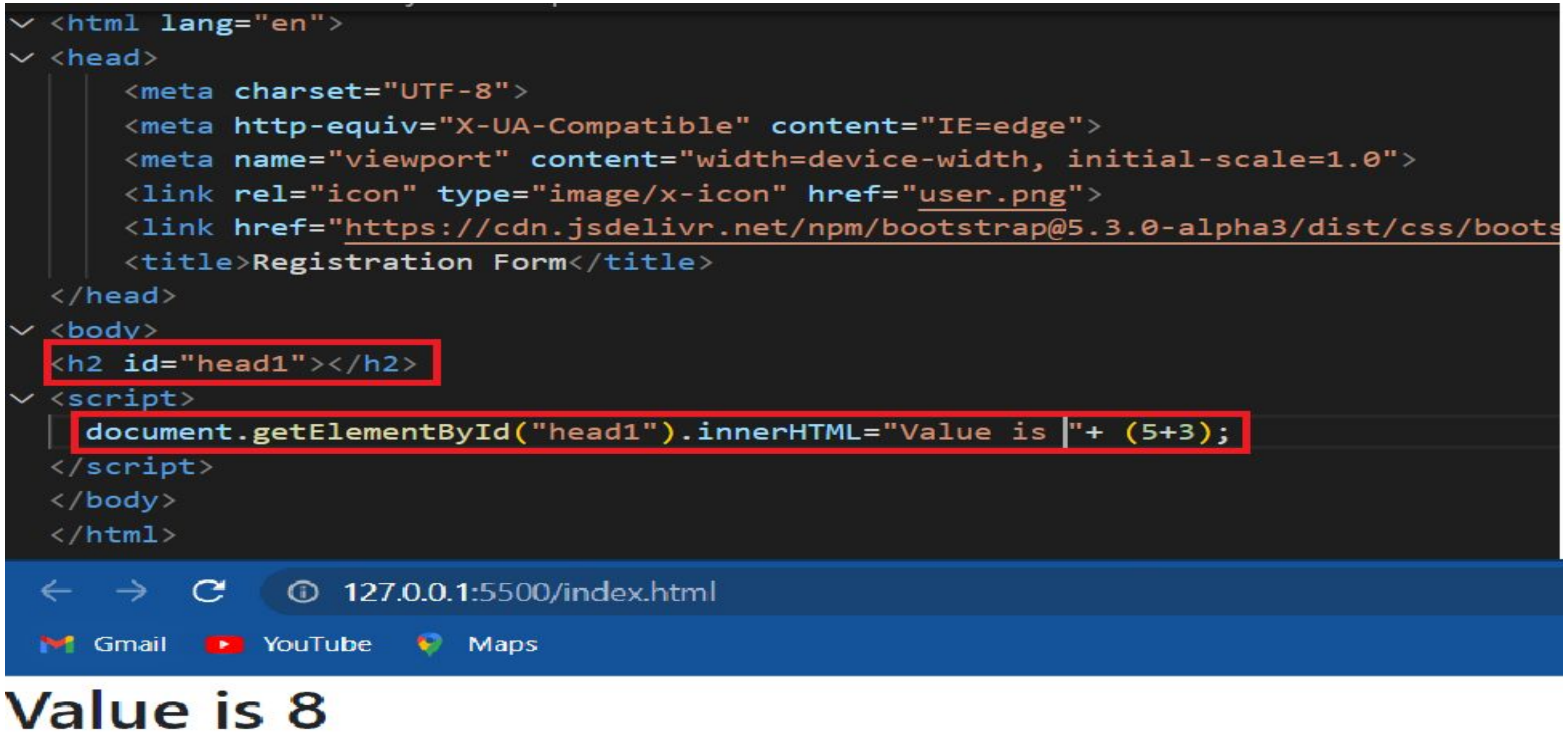
document.querySelector(): This method allows you to select elements using CSS selector syntax. It returns the first element that matches the specified selector.

Where to Display JS output?

JavaScript can "display" data in different ways:

- Writing into an HTML element, using innerHTML
- Writing into the HTML output using document.write()
- Writing into an alert box, using window.alert()
- Writing into the browser console, using console.log()

By Using innerHTML

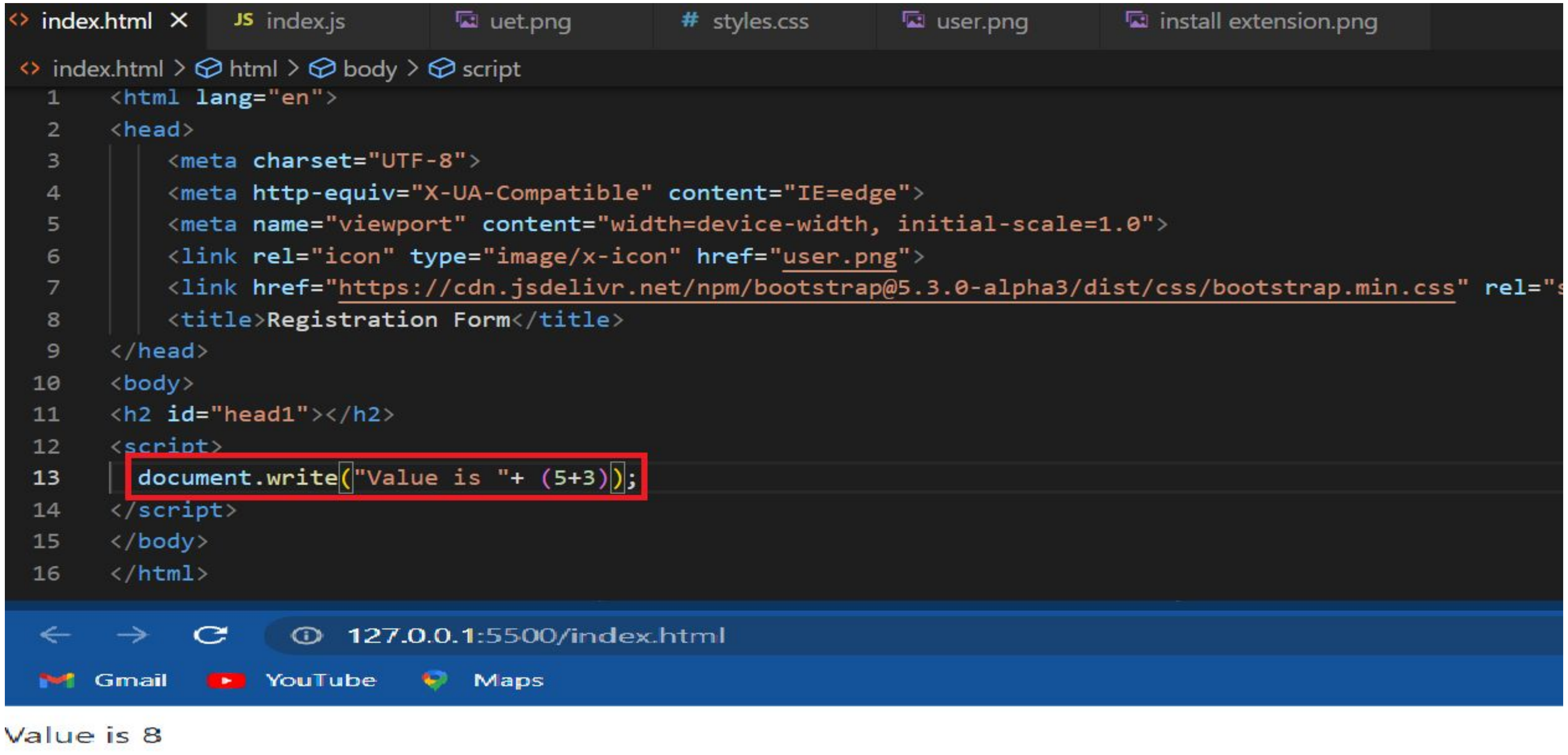


```
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="icon" type="image/x-icon" href="user.png">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/boots
    <title>Registration Form</title>
</head>
<body>
<h2 id="head1"></h2>
<script>
  document.getElementById("head1").innerHTML="Value is "+ (5+3);
</script>
</body>
</html>
```

127.0.0.1:5500/index.html

Value is 8

By Using document.write

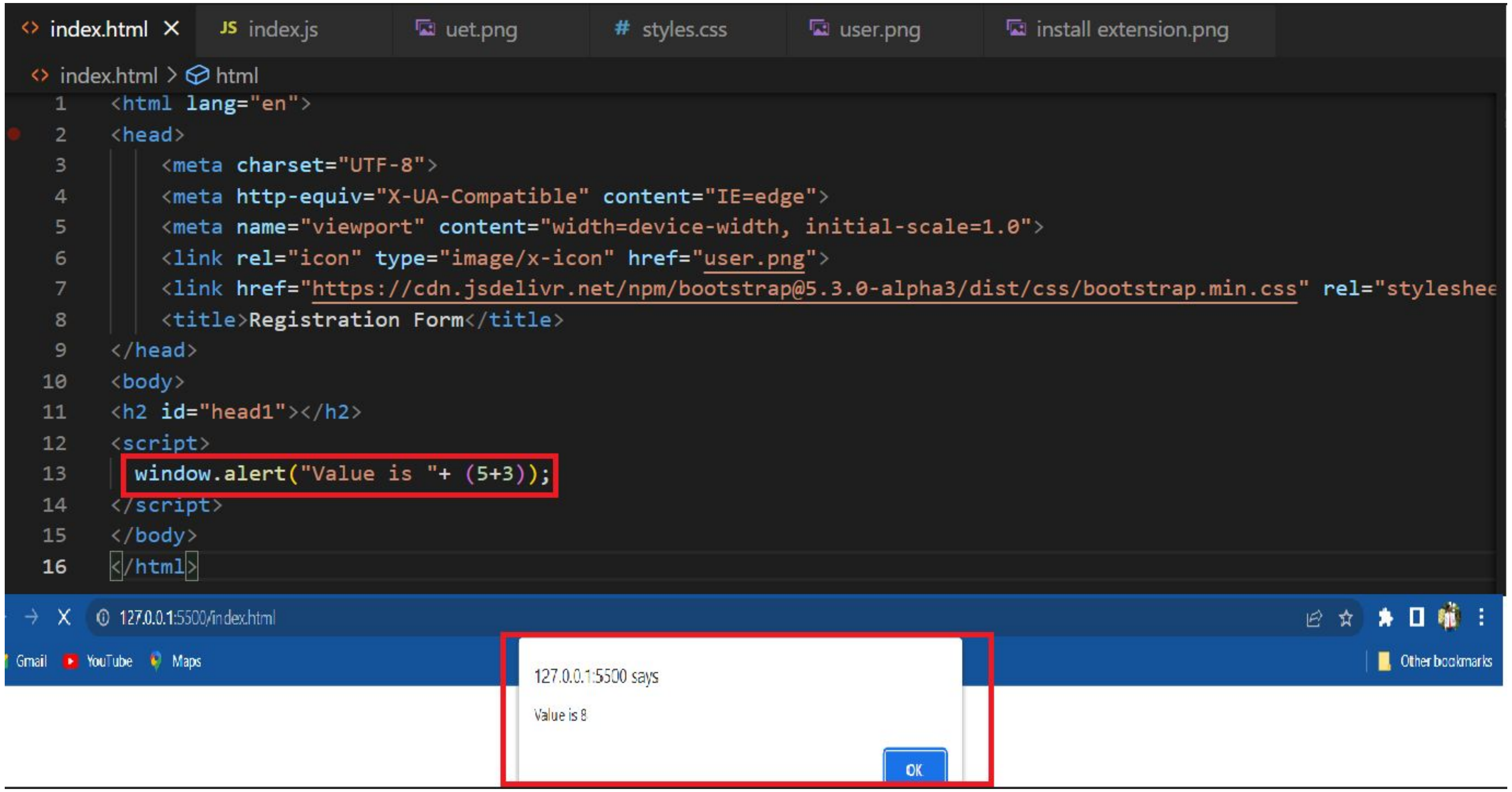


```html
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="icon" type="image/x-icon" href="user.png">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css" rel="s
    <title>Registration Form</title>
</head>
<body>
<h2 id="head1"></h2>
<script>
 document.write("Value is "+ (5+3));
</script>
</body>
</html>
```

127.0.0.1:5500/index.html

```
Value is 8
```

By Using window.alert



```
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="icon" type="image/x-icon" href="user.png">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css" rel="styleshee
    <title>Registration Form</title>
</head>
<body>
<h2 id="head1"></h2>
<script>
  window.alert("Value is "+ (5+3));
</script>
</body>
</html>
```

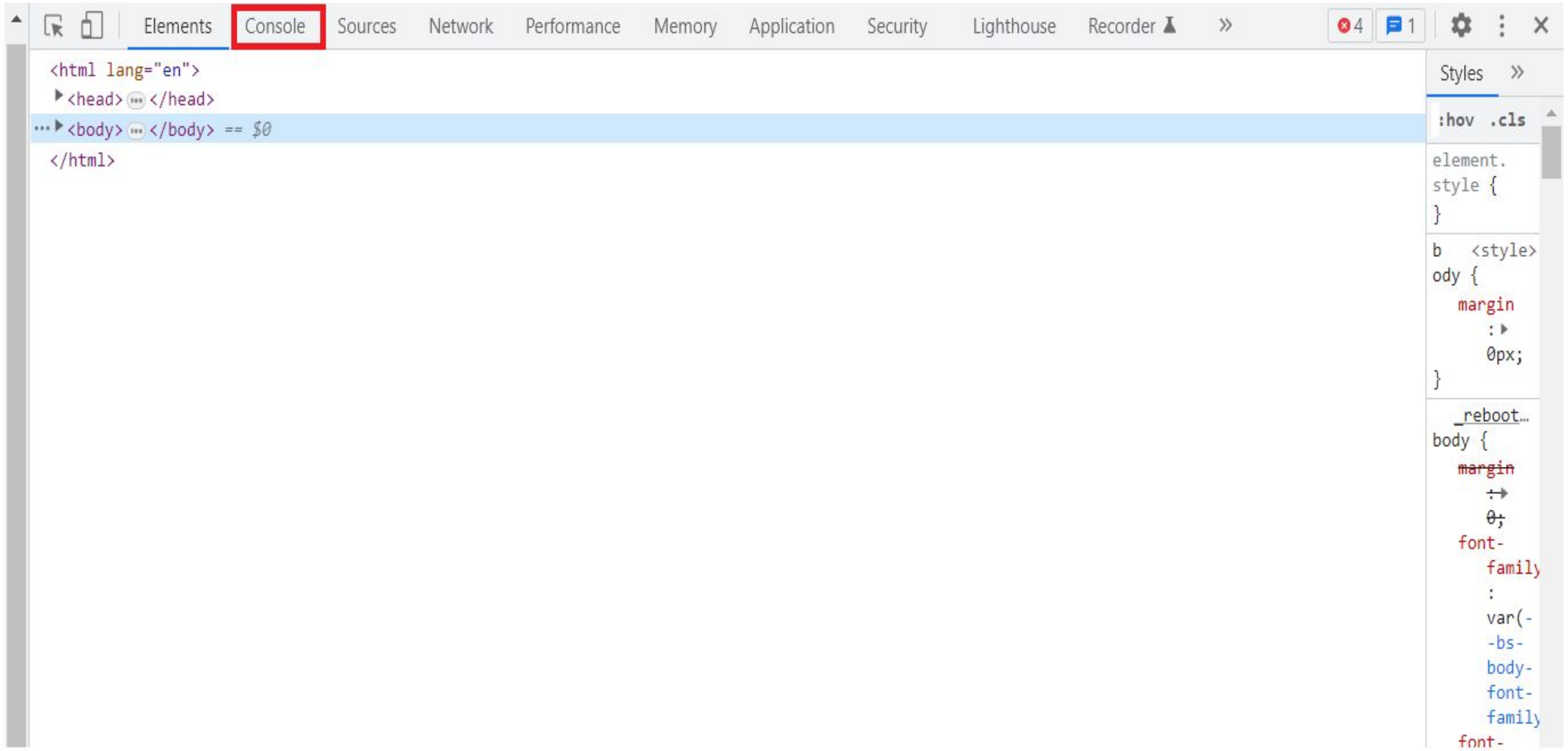
127.0.0.1:5500/index.html

127.0.0.1:5500 says

Value is 8

By Using console.log

To open console write click on chrome screen and press **inspect** after this you will see tabs on window press console tab.



```
<html lang="en">
  ▶<head>…</head>
… ▶<body>…</body> == $0
</html>
```

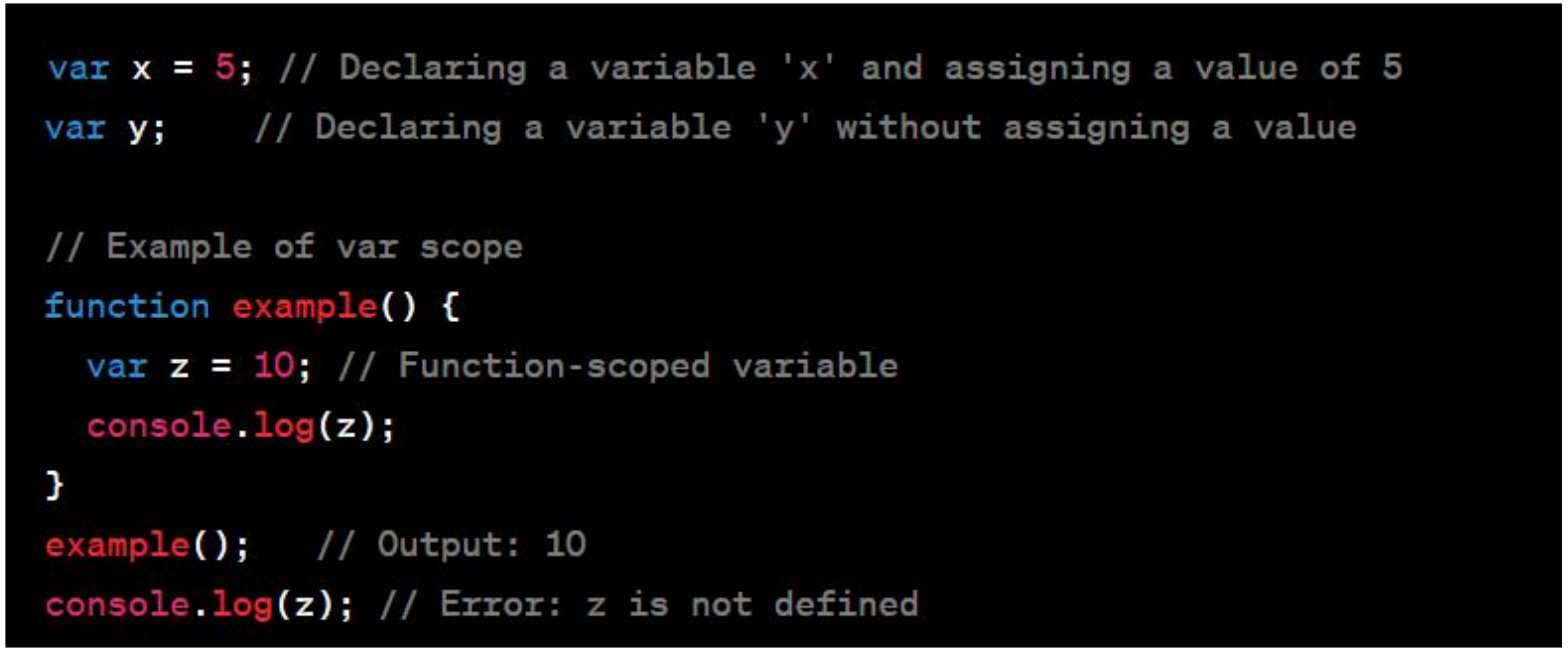
```
element.
style {
}
b    <style>
ody {
   margin
      : ▶
      0px;
}
  _reboot…
body {
   ~~margin~~
      ~~: ▶~~
      ~~0;~~
   font-
      family
      :
      var(-
      -bs-
      body-
      font-
      family
   font-
```

Variables in JS

In JavaScript, variables are used to store and manipulate data. They are declared using the **var, let,** or **const** keywords, followed by the variable name. Here's a breakdown of each type of variable declaration

- **var:** The var keyword is used to declare variables with function scope or global scope. Variables declared with var are hoisted, meaning they are moved to the top of their scope. They can be redeclared and reassigned.

Var Variable

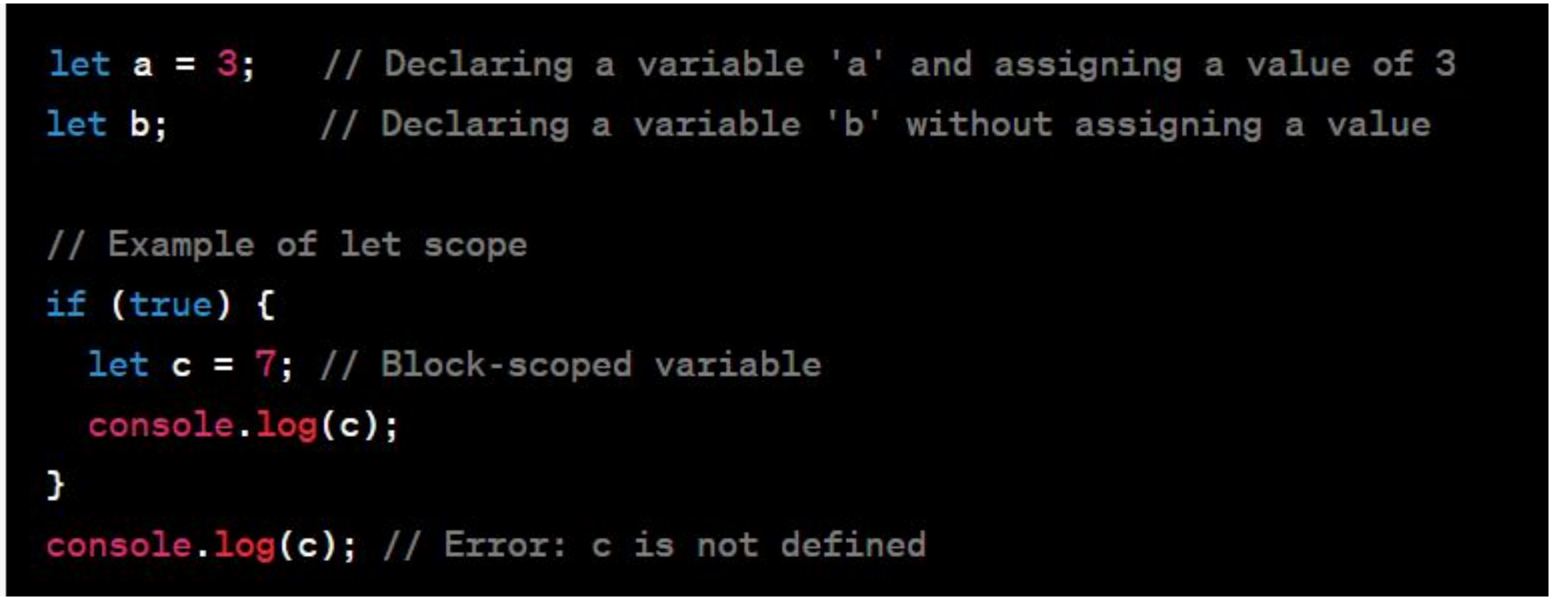


```
var x = 5; // Declaring a variable 'x' and assigning a value of 5
var y;    // Declaring a variable 'y' without assigning a value

// Example of var scope
function example() {
  var z = 10; // Function-scoped variable
  console.log(z);
}
example();   // Output: 10
console.log(z); // Error: z is not defined
```

- **let:** The let keyword was introduced in ES6 (ECMAScript 2015) and is used for block-scoped variables. Variables declared with let have block-level scope, meaning they are limited to the nearest enclosing block (inside curly braces {}). They can be reassigned but not redeclared in the same scope.

let Variable

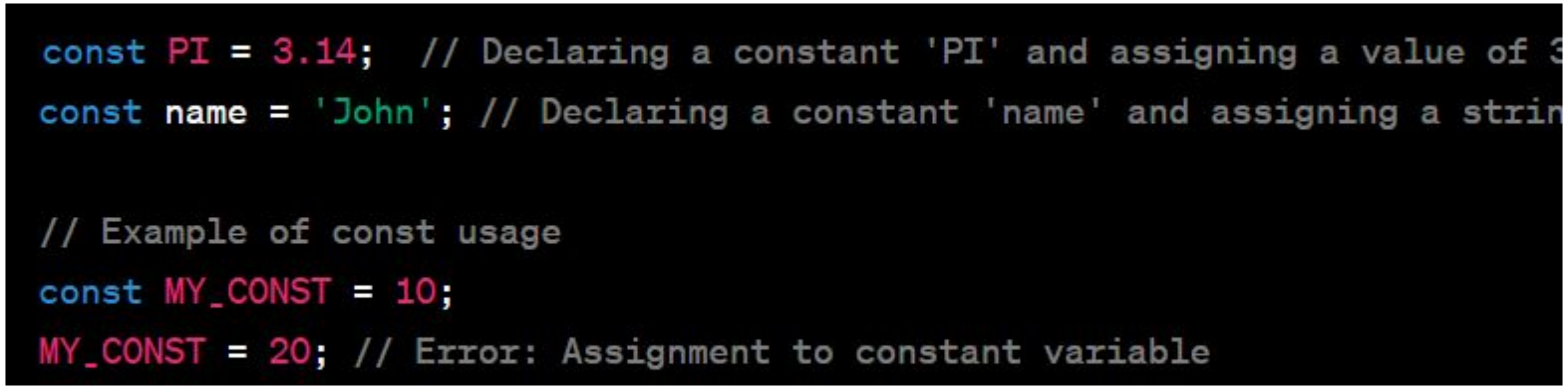


```
let a = 3;    // Declaring a variable 'a' and assigning a value of 3
let b;        // Declaring a variable 'b' without assigning a value

// Example of let scope
if (true) {
  let c = 7; // Block-scoped variable
  console.log(c);
}
console.log(c); // Error: c is not defined
```

- **const:** The const keyword is used to declare constants, which are variables that cannot be reassigned after they are defined. Like let, const is also block-scoped

const Variable



```
const PI = 3.14;  // Declaring a constant 'PI' and assigning a value of 3
const name = 'John'; // Declaring a constant 'name' and assigning a strin

// Example of const usage
const MY_CONST = 10;
MY_CONST = 20; // Error: Assignment to constant variable
```

Operators in Javascript (Arithmatic)

Addition: +

Subtraction: -

Multiplication: *

Division: /

Remainder (Modulus): %

Increment: ++

Decrement: --

Operators in Javascript (Assignment)

Assignment: =

Addition assignment: +=

Subtraction assignment: -=

Multiplication assignment: *=

Division assignment: /=

Remainder assignment: %=

Operators in Javascript (Comparison)

Equal to: ==

Not equal to: !=

Greater than: >

Less than: <

Greater than or equal to: >=

Less than or equal to: <=

Operators in Javascript (Logical)

Logical AND: &&

Logical OR: ||

Logical NOT: !

Operators in Javascript (Bitwise)

Bitwise AND: &

Bitwise OR: |

Bitwise XOR: ^

Bitwise NOT: ~

Left shift: <<

Right shift: >>

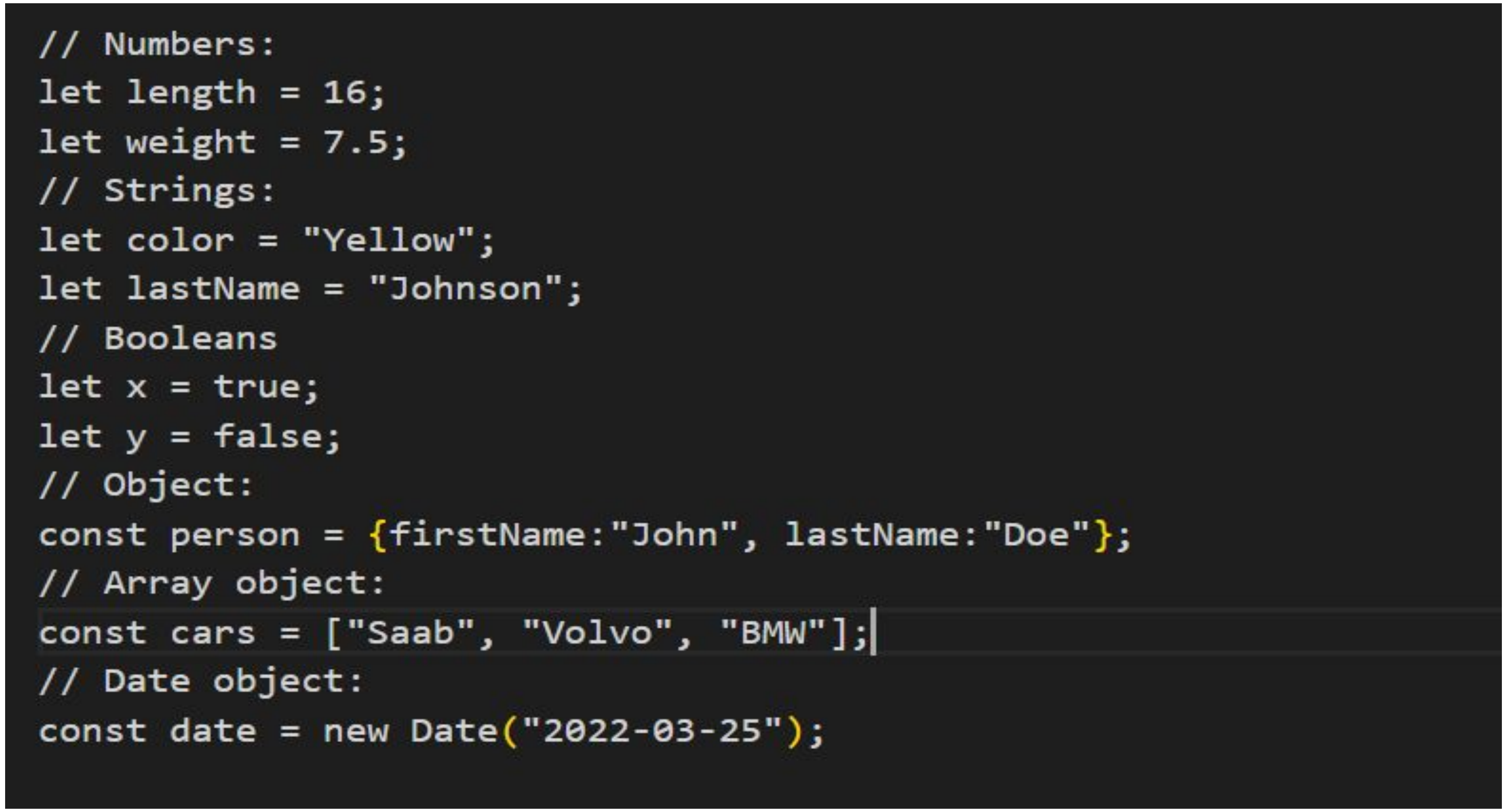
Zero-fill right shift: >>>

Datatypes in JS

Datatypes in javascript

- String
- Number
- Bigint
- Boolean
- Undefined
- Null
- Object (Can contain object,array or date)

Datatypes in JS (Examples)



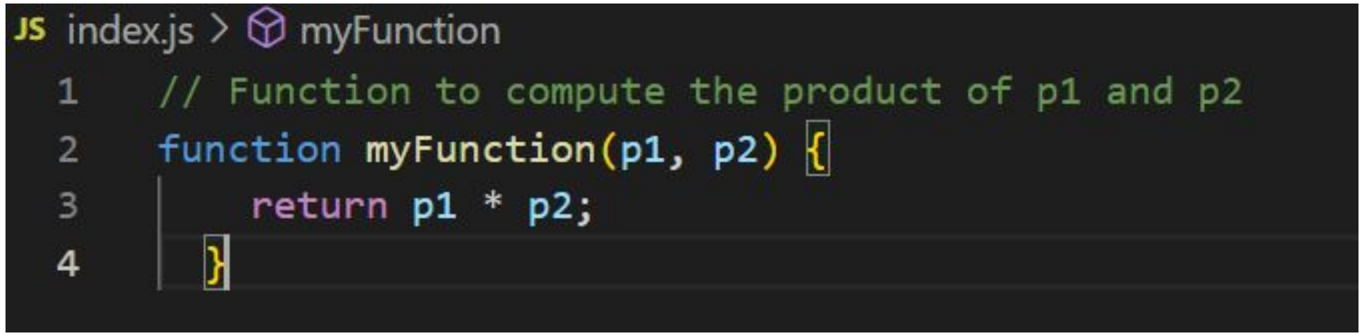
```
// Numbers:
let length = 16;
let weight = 7.5;
// Strings:
let color = "Yellow";
let lastName = "Johnson";
// Booleans
let x = true;
let y = false;
// Object:
const person = {firstName:"John", lastName:"Doe"};
// Array object:
const cars = ["Saab", "Volvo", "BMW"];
// Date object:
const date = new Date("2022-03-25");
```

Functions in JS

A JavaScript function is a block of code designed to perform a particular task. A JavaScript function is executed when "something" invokes it (calls it).

Function invokes when

- When an event occurs (when a user clicks a button)
- When it is invoked (called) from JavaScript code
- Automatically (self invoked)



JS index.js > myFunction

```javascript
// Function to compute the product of p1 and p2
function myFunction(p1, p2) {
    return p1 * p2;
}
```

Javascript Objects

In real life, a car is an **object**.

A car has **properties** like weight and color, and **methods** like start and stop:

const car = {type:"Fiat", model:"500", color:"white"};

Object	Properties	Methods
	car.name = Fiat	car.start()
	car.model = 500	car.drive()
	car.weight = 850kg	car.brake()
	car.color = white	car.stop()

Javascript Events

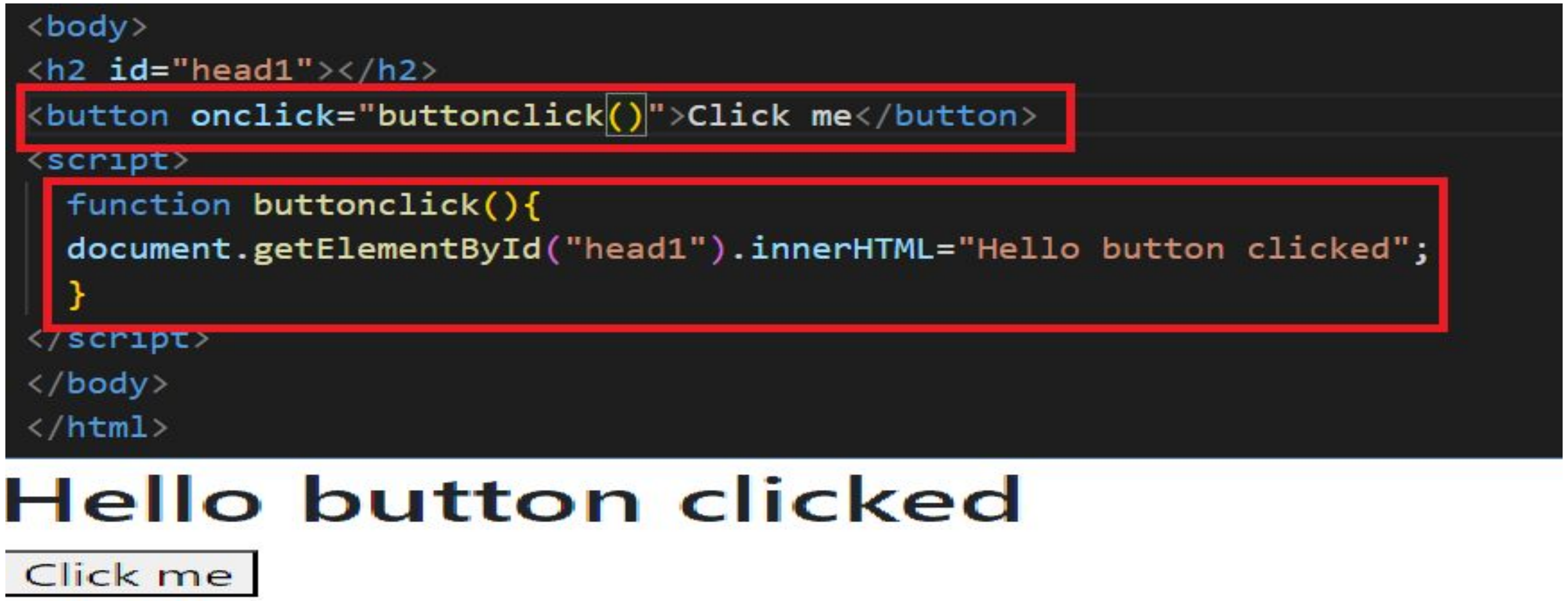
An HTML event can be something the browser does, or something a user does.

Here are some examples of HTML events:

- An HTML web page has finished loading
- An HTML input field was changed
- An HTML button was clicked

Javascript Events (Example)

When user click on button click me onclick event is invoked and the text (Hello button clicked) will show.



```
<body>
<h2 id="head1"></h2>
<button onclick="buttonclick()">Click me</button>
<script>
  function buttonclick(){
  document.getElementById("head1").innerHTML="Hello button clicked";
  }
</script>
</body>
</html>
```

Hello button clicked

Click me

Important JS Events

Mouse Events

- onclick
- ondblclick
- onmouseover
- onmouseout
- onmousemove
- onmousedown
- Onmouseup

Keyboard Events

- onkeydown
- onkeyup
- onkeypress

Important JS Events

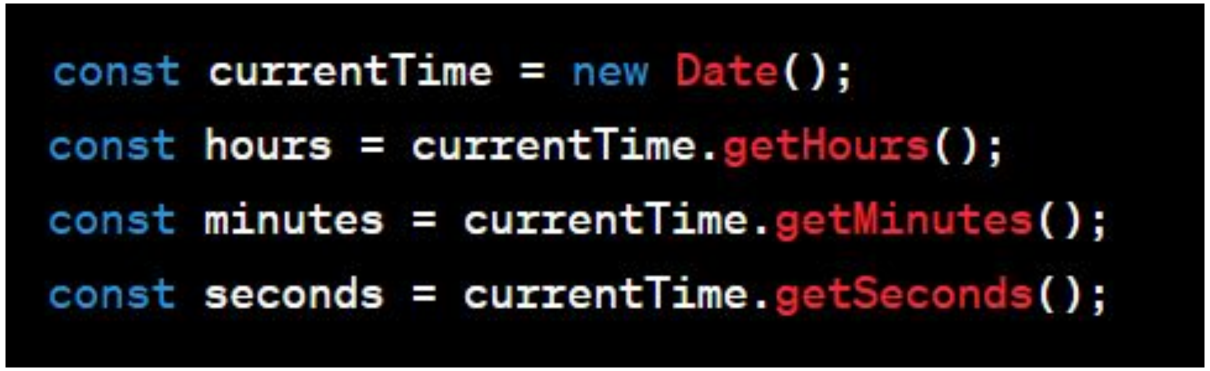
Form Events

- Onsubmit - The onsubmit event occurs when a form is submitted
- Onreset - The onreset event is triggered when a form is reset (clear)
- Onfocus - The onfocus event occurs when an element receives focus, typically when a user interacts with it, such as clicking inside an input field or navigating to it using the keyboard
- Onblur - The onblur event is the opposite of onfocus. It is triggered when an element loses focus, typically when a user moves away from it or clicks outside of it.
- Onchange - The onchange event occurs when the value of an element has changed

Make Clock using JS

Lets do a task in javascript by making a clock. First of all make a HTML file and a javascript file.

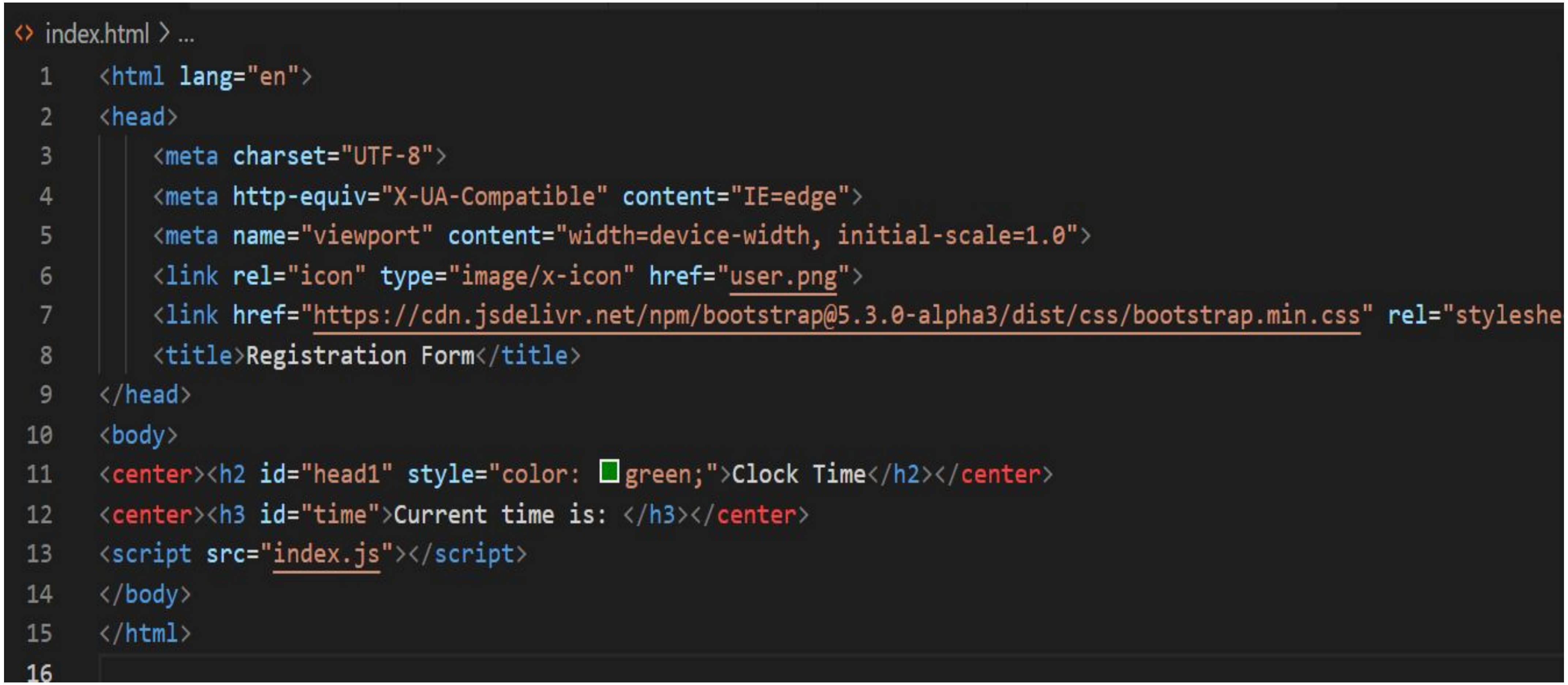
To get date in javascript we use



```
const currentTime = new Date();
const hours = currentTime.getHours();
const minutes = currentTime.getMinutes();
const seconds = currentTime.getSeconds();
```

Make Clock using JS

Here is HTML code



index.html > ...

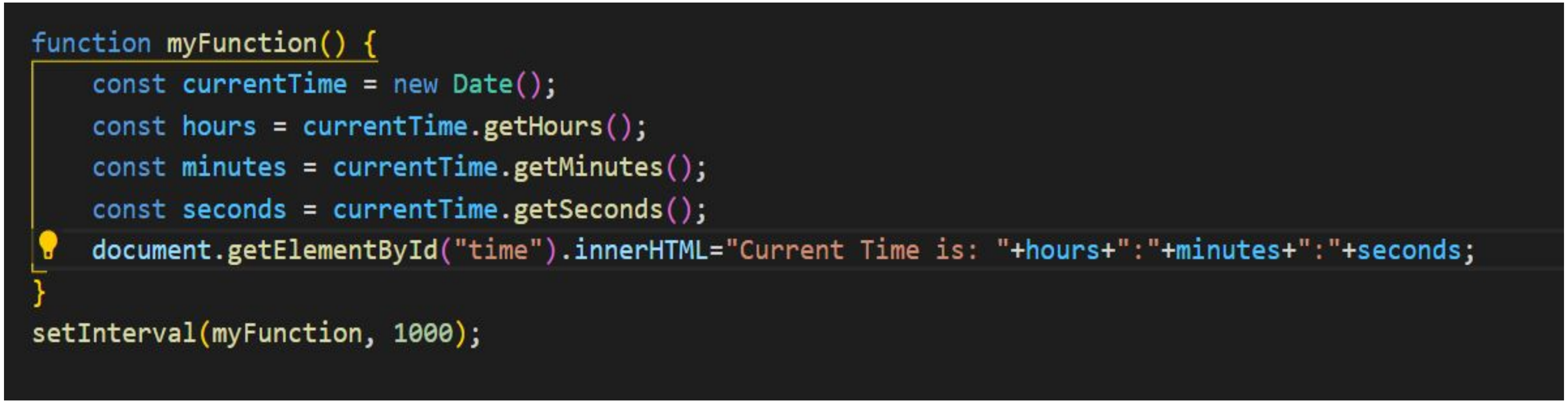
```html
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="icon" type="image/x-icon" href="user.png">
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css" rel="styleshe
    <title>Registration Form</title>
</head>
<body>
<center><h2 id="head1" style="color: 🟩green;">Clock Time</h2></center>
<center><h3 id="time">Current time is: </h3></center>
<script src="index.js"></script>
</body>
</html>

```

Make Clock using JS

Here is JS code. SetInteval is the time interval function and its syntax is

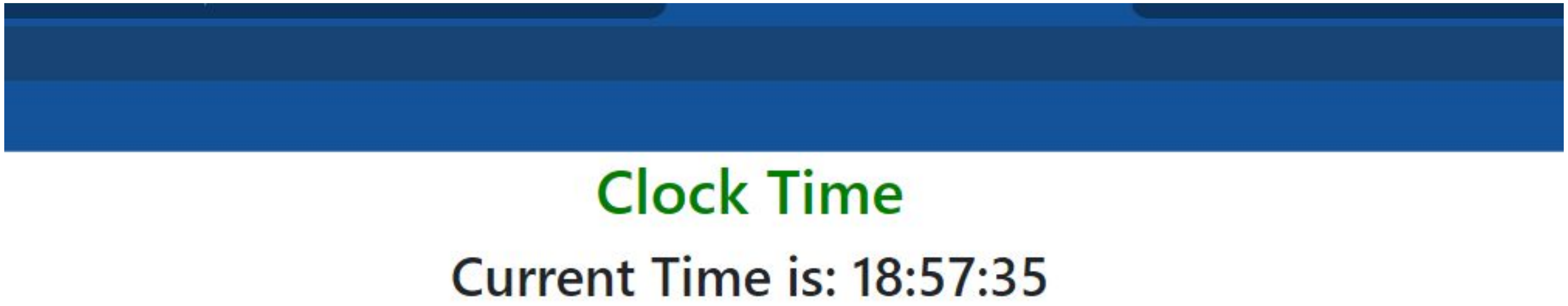
setInterval(code to execute, time after which invoke)



```
function myFunction() {
    const currentTime = new Date();
    const hours = currentTime.getHours();
    const minutes = currentTime.getMinutes();
    const seconds = currentTime.getSeconds();
    document.getElementById("time").innerHTML="Current Time is: "+hours+":"+minutes+":"+seconds;
}
setInterval(myFunction, 1000);
```

Make Clock using JS

Here is the output



```
Clock Time
Current Time is: 18:57:35
```

Class # 1 Task

Create an HTML page for registration of university students for this take input of

- Student Name
- Roll No.
- Matric Marks
- Fsc Marks
- Entry test Marks

Below these fields there is a submit button when user clicks on submit button show the student name and its aggregate on below of these fields.Calculate aggragate by adding 20% Matric marks, 30% FSC marks and 50% entry test marks.

If Student Aggregate is greater than 60% show alert you are eligible for admission if less than 60% then show alert you are not eligible.