SIX WEEKS SUMMER TRAINING REPORT

On

Full Stack Web Development with NodeJS

Submitted by:

Shivankit Raghav

11908193

B. Tech CSE

Under the guidance of

Coding Blocks

School of Computer Science and Engineering
Lovely Professional University, Phagwara

(June- July 2021)

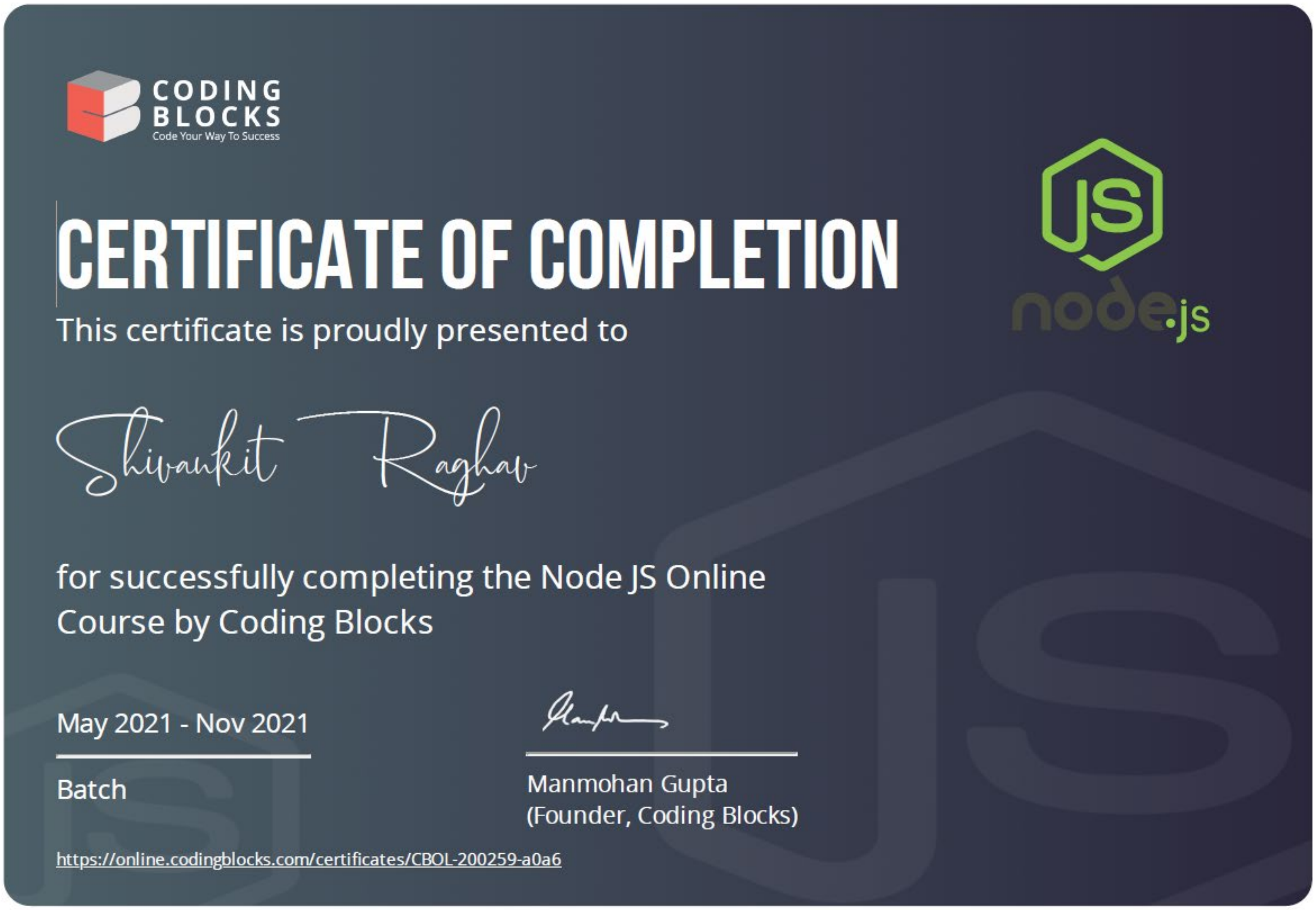
DECLARATION

I hereby declare that I have complete my six weeks summer training at CODING BLOCKS from 10th May to 15th July under the guidance of Mr. Arnav Gupta. I hereby declare that I have worked with full dedication during these six weeks of training and my learning outcomes fulfill the requirements for the award of degree of B. Tech CSE, Lovely Professional University, Phagwara.

Shivankit Raghav

11908193

Date: 20 September 2021


CODING BLOCKS
Code Your Way To Success

CERTIFICATE OF COMPLETION

This certificate is proudly presented to

Shivankit Raghav

for successfully completing the Node JS Online Course by Coding Blocks

May 2021 - Nov 2021

Batch

[illegible]

Manmohan Gupta
(Founder, Coding Blocks)

https://online.codingblocks.com/certificates/CBOL-200259-a0a6

ACKNOWLEGMENT

I would like to express my gratitude towards my University as well as Coding Blocks for providing me the golden opportunity to do this wonderful summer training regarding Web Development, which helped me to learn a lot. As a result, I came to know about so many new things. So, I am really thankful to them. Moreover, I would like to thank my friends who helped me a lot whenever I got stuck in some problem related to my course. I am really thankful to have such a good support of them as they always have my back whenever I need. Also, I would like to mention the support system and consideration of my parents who have always been there in my life to make me choose right thing and oppose the wrong. Without them I could never had learned and became a person who I am now. I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thank you to all of them.

INTRODUCTION

Web development is a broad term for the work involved in developing a web site for the Internet (World Wide Web) or an intranet (a private network). Web development can range from developing the simplest static single page of plain text to the most complex web-based internet applications (or just 'web apps') electronic businesses, and social network services. A more comprehensive list of tasks to which web development commonly refers, may include web engineering, web design, web content development, client-side/server-side scripting, web server and network security configuration, and e-commerce development.

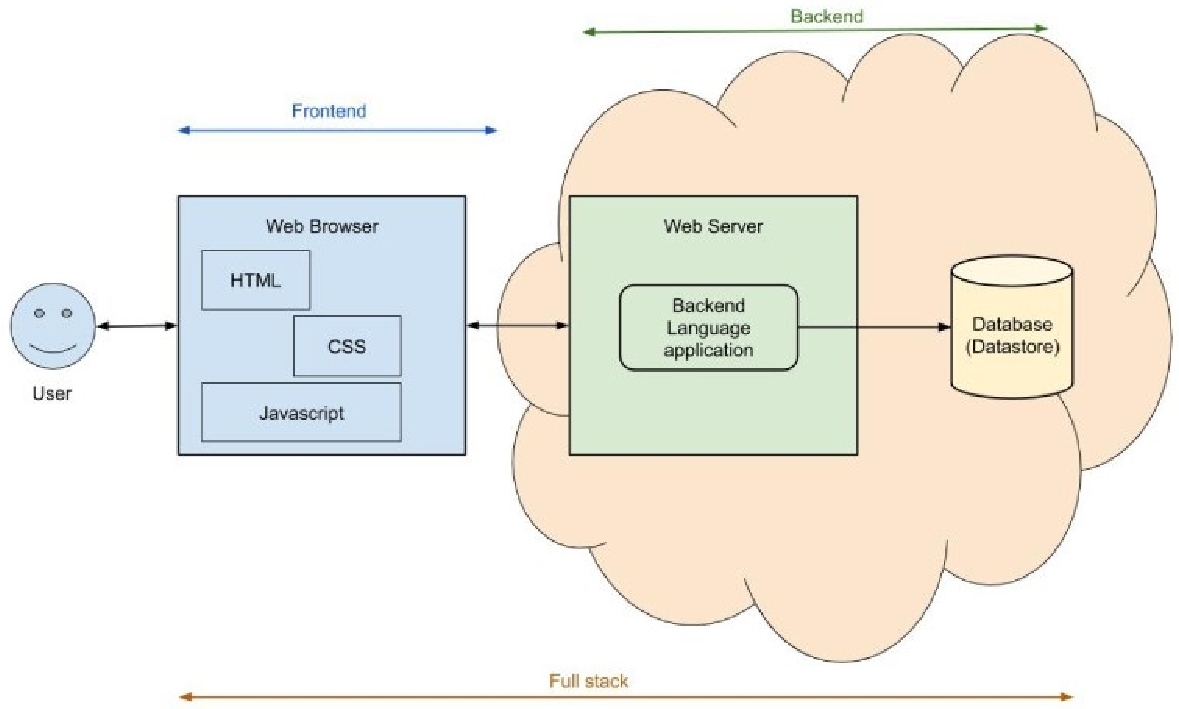
For larger organizations and businesses, web development teams can consist of hundreds of people (web developers) and follow standard methods like Agile methodologies while developing websites. Smaller organizations may only require a single permanent or contracting developer, or secondary assignment to related job positions such as a graphic designer or information systems technician. Web development may be a collaborative effort between departments rather than the domain of a designated department. There are three kinds of web developer specialization: front-end developer, back-end developer, and full-stack developer.

Front end vs Back end

Front End	Back End
The front end is the presentation layer of a piece of software. It is what the user can see and directly interact with. It is the front-end developer's job to create the user-facing elements of the browser or app, and produce an environment for everything that the user interacts with. They write code that brings the site's design to life and enables the functionality created by the Back End Developer.	The Back end is the server side of an application that users don't directly interact with. This includes everything that communicates between an application's database and the browser or app. The back-end developer writes code to ensure data is stored and accessed securely. They create the functionality that works behind the scenes to enable the front end features that the users interact with.
HTML CSS JavaScript Angular React Bootstrap	Java Node.js Python mySQL Mongo Firebase

What is a Full Stack developer then?

Full stack is a combination of front and back-end development. Though everyone's experience will vary, Full stack developers tend to have a degree of familiarity and understanding with the entire software stack. Many hiring managers make the mistake of thinking they want a full stack developer when what they actually need is two people – a front-end developer and a back-end developer. One person will never be an expert in everything, but a full stack developer will know enough to help coordinate the project, talk and learn what they need to know in order to deliver the project.



Skills Required:

For being a successful web developer, one should possess the following skills:

- Understanding of client and server-side scripting.
- Creating, editing and modifying templates for a web development framework.
- Testing cross browser inconsistencies.
- Conducting observation user testing.
- Testing for compliance to specified standards such as accessibility standards in the client region

Along with these, one more important thing for web developer is communication. Because a web developer is always in contact with other people. A front-end developer will stay in contact with web designers and back end developers while the backend developer will stay in contact with front end developer and data base managers. So it’s important for web developers to have good communication skills so that they are able to express themselves properly.

Technologies Learnt

During the period of 6 weeks, I have learnt various Web development technologies, some of which are used for front end while others are used for back end. All these languages together combined results in formation of web pages. A brief introduction of all these technologies is provided below:

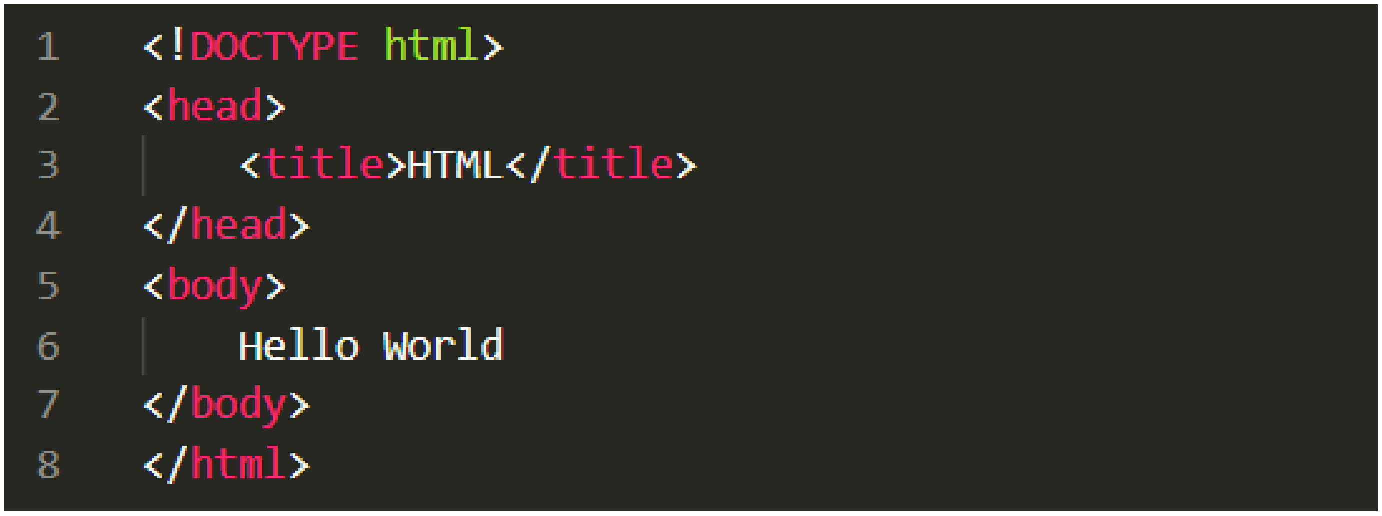
FRONT END:

1. HTML

HyperText Markup Language, or **HTML** is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript. Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by *tags*, written using angle brackets. Tags such as <img /> and <input />

directly introduce content into the page. Other tags such as <p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page. HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content.

A simple HTML program to print "Hello World"



```
<!DOCTYPE html>
<head>
    <title>HTML</title>
</head>
<body>
    Hello World
</body>
</html>
```

All HTML pages have a series of **HTML elements**, consisting of a set of **tags** and **attributes**. HTML elements are the building blocks of a web page. A tag tells the web browser where an element begins and ends, whereas an attribute describes the characteristics of an element.

The three main parts of an element are:

- **Opening tag –** used to state where an element starts to take effect. The tag is wrapped with opening and closing angle brackets. For example, use the start tag **<p>** to create a paragraph.
- **Content –** this is the output that other users see.
- **Closing tag** – the same as the opening tag, but with a forward slash before the element name. For example, **</p>** to end a paragraph.

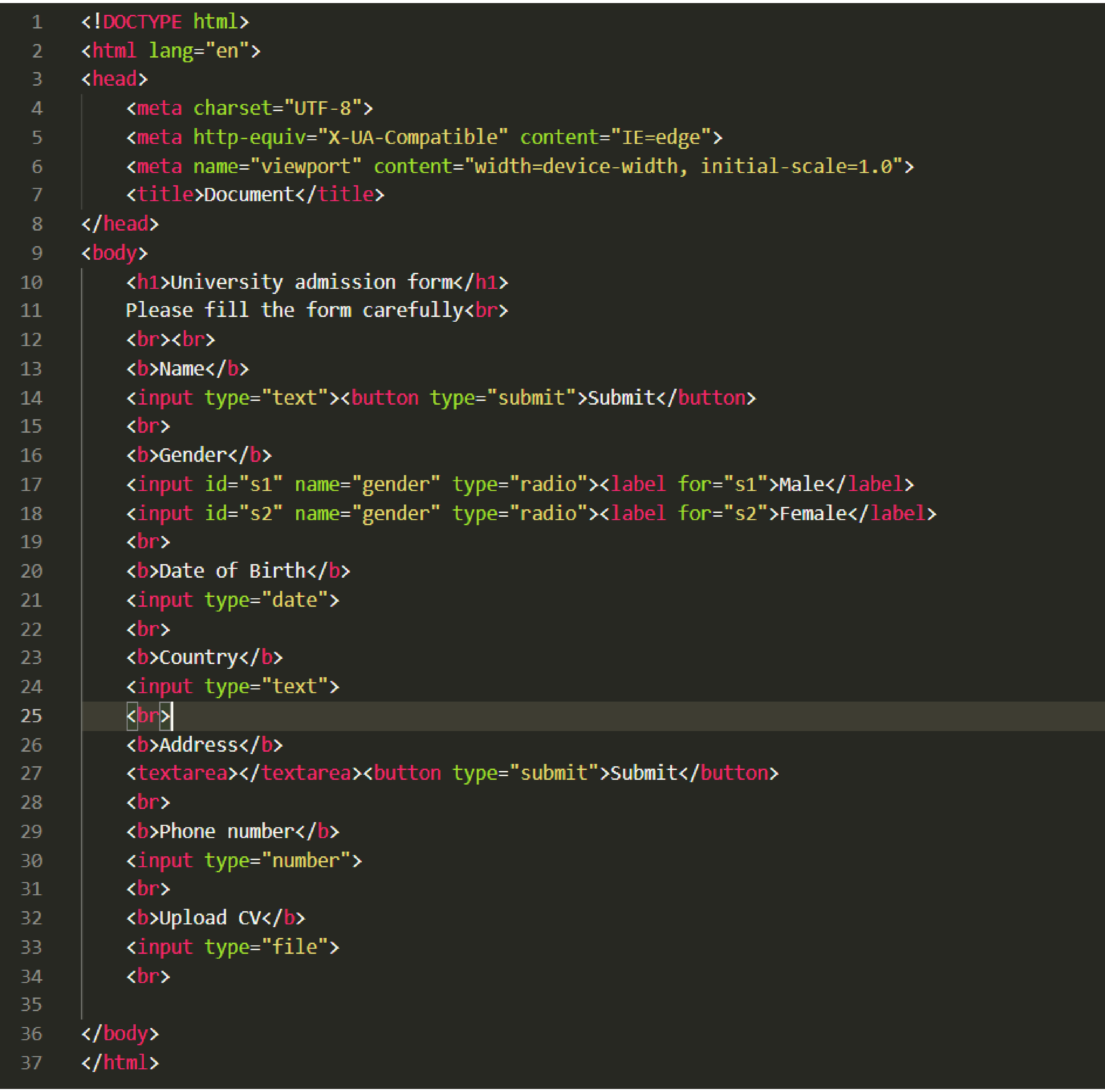
The combination of these three parts will create an HTML element:

<p>This is how you add a paragraph in HTML.</p>

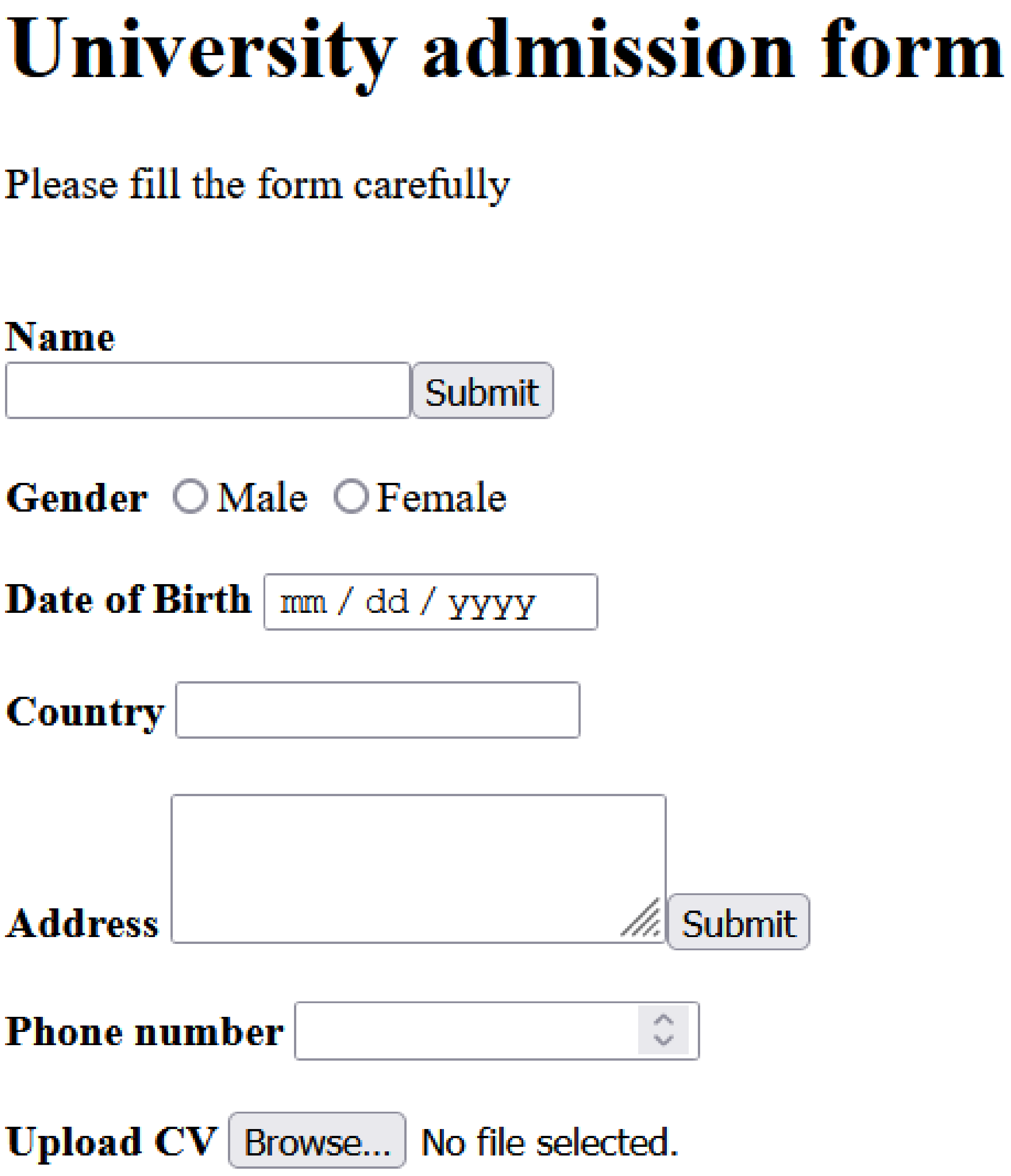
Another critical part of an HTML element is its **attribute**, which has two sections – a **name** and **attribute value**. The name identifies the additional information that a user wants to add, while the attribute value gives further specifications.

Throughout the course, HTML served as the bread on which we tried our recipes.

One such example is this code that I wrote and then its output:



```html
<!DOCTYPE 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">
    <title>Document</title>
</head>
<body>
    <h1>University admission form</h1>
    Please fill the form carefully<br>
    <br><br>
    <b>Name</b>
    <input type="text"><button type="submit">Submit</button>
    <br>
    <b>Gender</b>
    <input id="s1" name="gender" type="radio"><label for="s1">Male</label>
    <input id="s2" name="gender" type="radio"><label for="s2">Female</label>
    <br>
    <b>Date of Birth</b>
    <input type="date">
    <br>
    <b>Country</b>
    <input type="text">
    <br>
    <b>Address</b>
    <textarea></textarea><button type="submit">Submit</button>
    <br>
    <b>Phone number</b>
    <input type="number">
    <br>
    <b>Upload CV</b>
    <input type="file">
    <br>

</body>
</html>
```



University admission form

Please fill the form carefully

Name

[ ] Submit

Gender ○ Male ○ Female

Date of Birth mm / dd / yyyy

Country [ ]

Address [ ] Submit

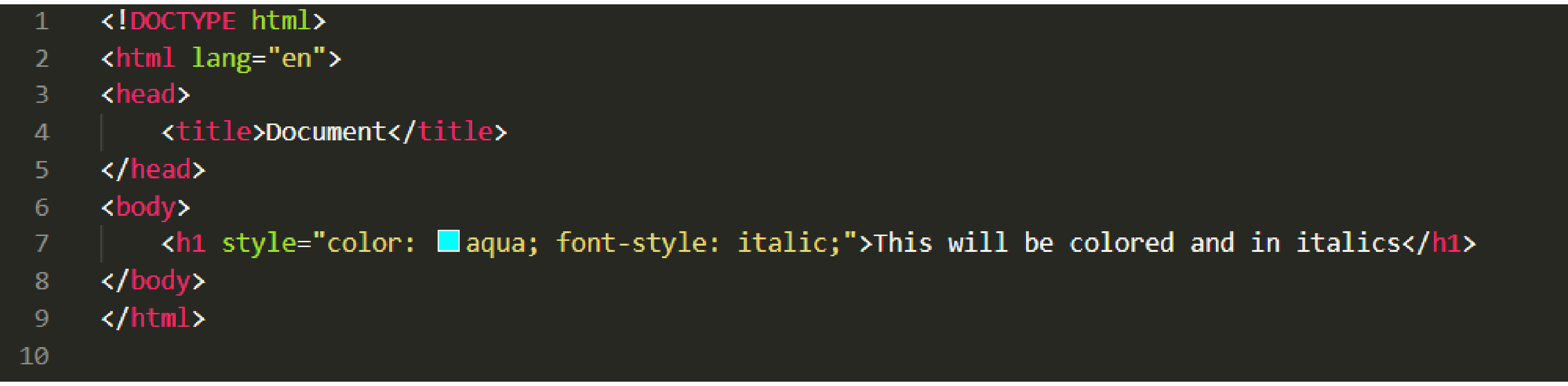
Phone number [ ]

Upload CV Browse... No file selected.

2. CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript. CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file which reduces complexity and repetition in the structural content as well as enabling the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

A simple CSS program to add color and style to basic html font:



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Document</title>
</head>
<body>
    <h1 style="color: aqua; font-style: italic;">This will be colored and in italics</h1>
</body>
</html>

```

And its output will be:

This will be colored and in italics

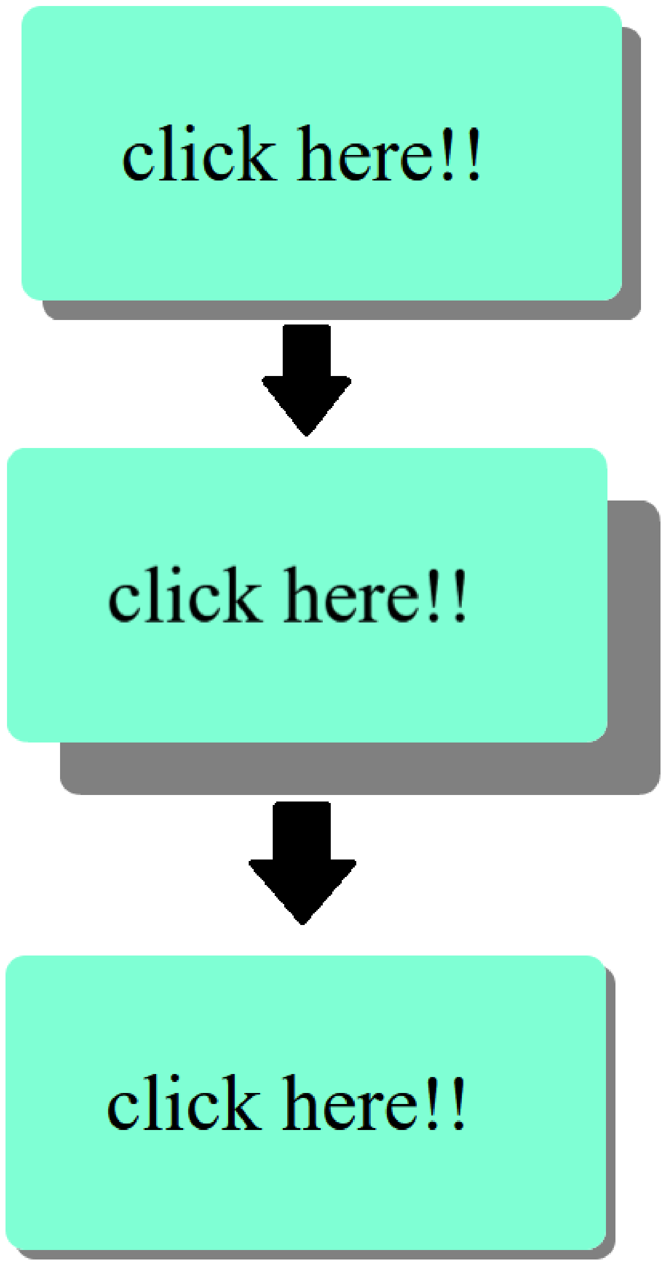
But this only shows the inline CSS. There are actually various ways to use CSS with your HTML code.

1. Inline CSS: An inline CSS is used to apply a unique style to a single HTML element. An inline CSS uses the ‘style’ attribute of an HTML element.

2. Internal CSS: An internal CSS is used to define a style for a single HTML page. An internal CSS is defined in the <head> section of an HTML page within a <style> element.

3. External CSS: An external CSS is used to define the style for many HTML pages. To use an external CSS, we add a link in the <head> section of each HTML page.

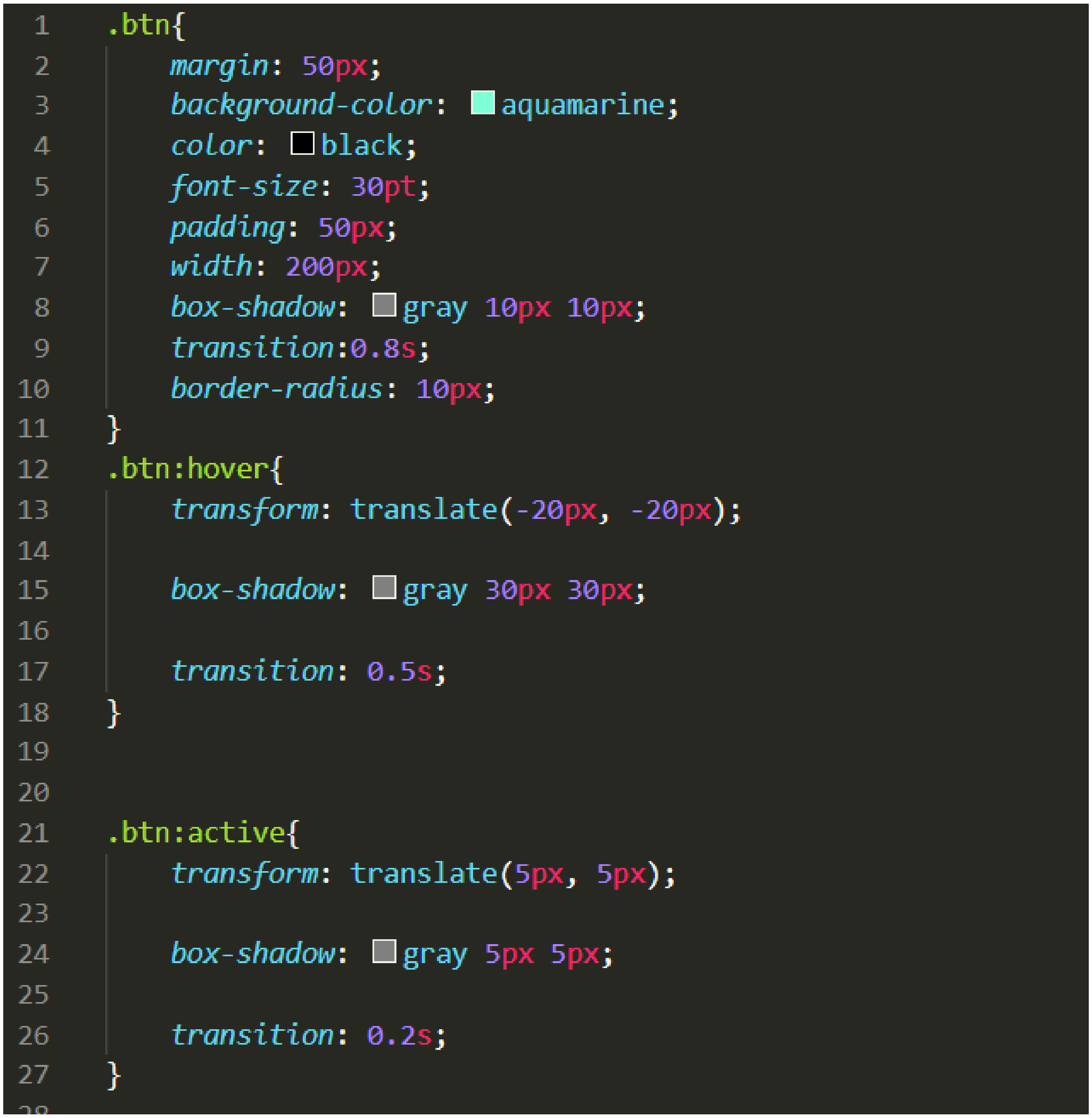
Below, I’ve shown a small button that I made in which I implemented more things from CSS



In the first state,
the button is in rest

When it is hovered
upon, it raises up

When it is clicked,
it goes down



```
.btn{
    margin: 50px;
    background-color: aquamarine;
    color: black;
    font-size: 30pt;
    padding: 50px;
    width: 200px;
    box-shadow: gray 10px 10px;
    transition:0.8s;
    border-radius: 10px;
}
.btn:hover{
    transform: translate(-20px, -20px);

    box-shadow: gray 30px 30px;

    transition: 0.5s;
}


.btn:active{
    transform: translate(5px, 5px);

    box-shadow: gray 5px 5px;

    transition: 0.2s;
}
```

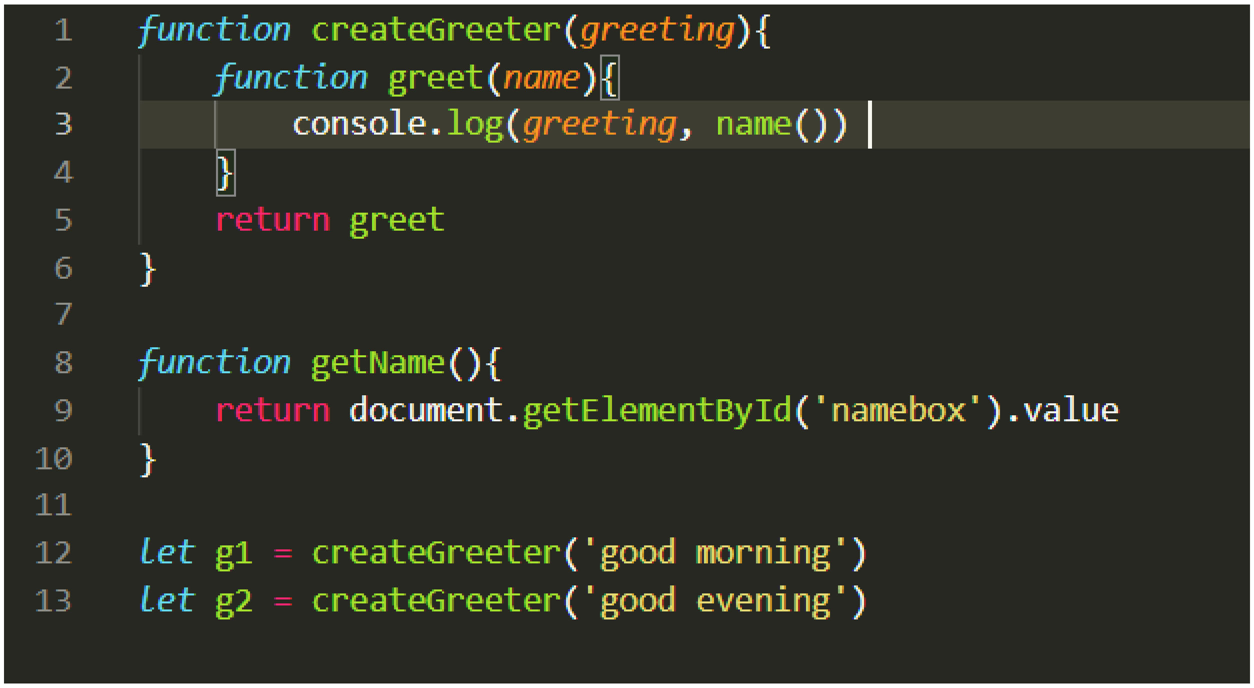
Code for the button shown above. In this code, .btn is the class which is given to the button. We've used various CSS attributes to get our desired results.

3. JavaScript

JavaScript (often shortened to **JS**) is a lightweight, interpreted, object-oriented language with first-class functions, and is best known as the scripting language for Web pages, but it's used in many non-browser environments as well. It is a prototype-based, multi-paradigm scripting language that is dynamic, and supports object-oriented, imperative, and functional programming styles. JavaScript runs on the client side of the web, which can be used to design / program how the web pages behave on the occurrence of an event. JavaScript is an easy to learn and also powerful scripting language, widely used for controlling web page behavior. JavaScript can function as both a procedural and an object-oriented language. Objects are created programmatically in JavaScript, by attaching methods and properties to otherwise empty objects **at run time**, as opposed to the syntactic class definitions common in compiled languages like C++ and Java. Once an object has been constructed it can be used as a blueprint (or prototype) for creating similar objects.

JavaScript's dynamic capabilities include runtime object construction, variable parameter lists, function variables, dynamic script creation (via eval), object introspection (via for ... in), and source code recovery (JavaScript programs can decompile function bodies back into their source text).

Javascript can also be used for normal programming questions like C++ and Java. It's not as efficient as them but it can still be used for those things.

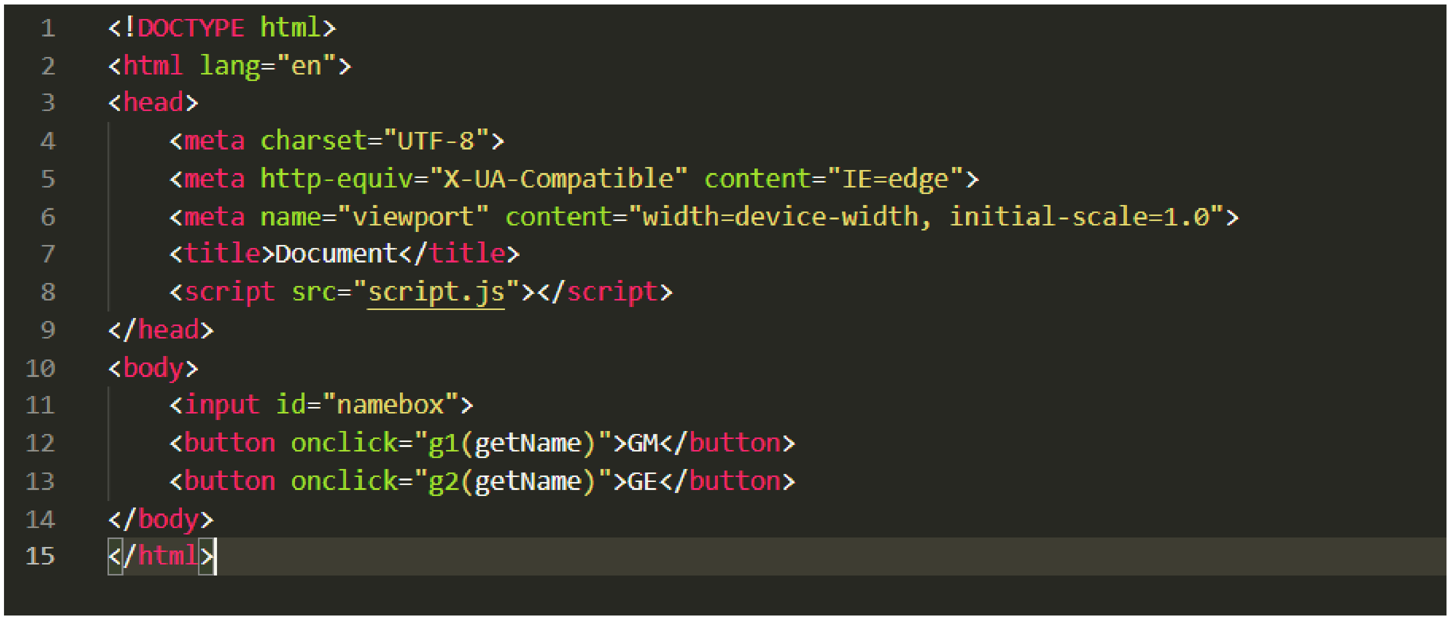


```
function createGreeter(greeting){
    function greet(name){
        console.log(greeting, name())
    }
    return greet
}

function getName(){
    return document.getElementById('namebox').value
}

let g1 = createGreeter('good morning')
let g2 = createGreeter('good evening')
```

One very interesting thing about java is the callback functions.

A callback function is a function passed into another function as an argument, which is then invoked inside the outer function to complete some kind of routine or action.

In the above code, createGreeter is a function that gets a 'greeting' parameter in it. In then has a greet function inside of it and that greet function is taking a parameter 'name' which is actually a function. So in this javascript files, 2 g1 and g2 objects are created that store the greet() function. When these objects get the name, they'll respond with the greeting that they have been given (g1 will give good morning and g2 will give good evening).



```html
<!DOCTYPE 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">
    <title>Document</title>
    <script src="script.js"></script>
</head>
<body>
    <input id="namebox">
    <button onclick="g1(getName)">GM</button>
    <button onclick="g2(getName)">GE</button>
</body>
</html>
```

So, from this html function, just suppose that GM is the button that will be clicked. So, when that button is clicked, it calls the g1 function with getname as the callback function. Then getname function will read the value from the input box and pass that name to the greet function which will then send the greeting and the name on the screen (Thus printing Good morning Shivankit) on the screen.

Callback functions are one of the most unique qualities of Javascript and they are incredibly useful. These are especially useful with things like Promises in which, we can pass a resolve function at a later stage when we want to deal with the results of the promise.

4. JQuery

jQuery is a fast, lightweight and feature rich Javascript library that is based on the principle “Write less, Do more”. It’s easy to use APIs make the things like HTML document traversal and manipulation, event handling, adding animation effects to a web page much simpler that works seamlessly across all the major browsers. jQuery also give you the ability to create an Ajax based application in a quick and simple way.

The biggest advantage of jQuery comes from its selectors that allow you to traverse the DOM tree of an HTML document’s structure in an efficient manner.

Additionally, using the jQuery inbuilt methods you can create animations and effects like sliding transition, showing or hiding element, etc. with a single line of code.

Save lots of time – You can save lots of time and efforts by using the jQuery inbuilt effects and selectors and concentrate on other development work.

Simply common JavaScript tasks – jQuery considerably simplifies the common JavaScript tasks. Now you can easily create feature rich and interactive web pages with fewer lines of codes, a typical example is retrieving the information from a server and updates the pages without refreshing.

Easy to use – jQuery is very easy to use. Anybody with the basic working knowledge of HTML, CSS and JavaScript can start development with jQuery.

Compatible with browsers – jQuery is created with modern browsers in mind and it is compatible with all major modern browsers such as Firefox, Chrome, Safari, Internet Explorer and Opera.

Absolutely Free – And the best part is, it is completely free to download and use.

jQuery reduce the work quite a bit and that's why, it's one of the most used libraries in the world.

If I use JavaScript,

```
document.getElementsByClassName("simple-li");
```

On the other hand, if I use jQuery,

```
$('simple-li')
```

So, the difference between the sizes of the syntax is quite evident from this.

BACK END:

5. Node.js

Node.js is an open-source, cross-platform, back-end JavaScript runtime environment that runs on the V8 engine and executes JavaScript code outside a web browser. Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser.

Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web-application development around a single programming language, rather than different languages for server-side and client-side scripts.

Though .js is the standard filename extension for JavaScript code, the name "Node.js" doesn't refer to a particular file in this context and is merely the name of the product. Node.js has an event-driven architecture capable of asynchronous I/O. These design choices aim to optimize throughput and scalability in web applications with many input/output operations, as well as for real-time Web applications (e.g., real-time communication programs and browser games).

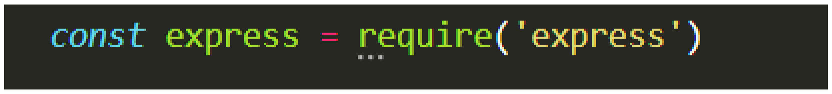
6. Express

Web Applications: Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.

APIs: With a myriad of HTTP utility methods and middleware at your disposal, creating a robust API is quick and easy.

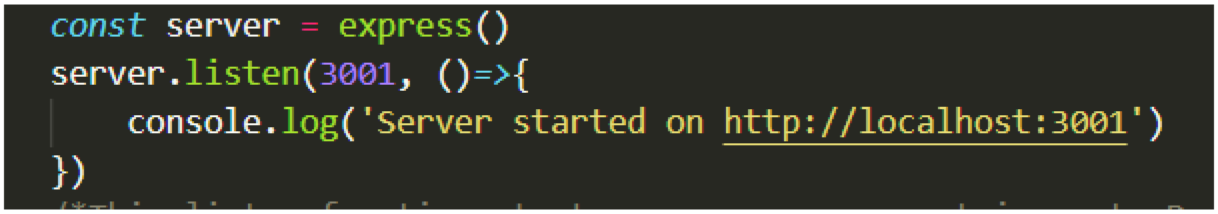
Performance: Express provides a thin layer of fundamental web application features, without obscuring Node.js features that you know and love.

To install express, write: $ npm install express –save



```
const express = require('express')
```

After installing express, we can bring it in our code by typing this. This runs up express and now we can write our backend code using it.



```
const server = express()
server.listen(3001, ()=>{
    console.log('Server started on http://localhost:3001')
})
```

This is basically how a server is started. The **server.listen() function** is used to bind and listen the connections on the specified host and port. This method is identical to Node's http.Server.listen() method.

If the port number is omitted or is 0, the operating system will assign an arbitrary unused port, which is useful for cases like automated tasks (tests, etc.).

We’ve said “localhost” because the server is currently running on our system only.

7. Sequelize

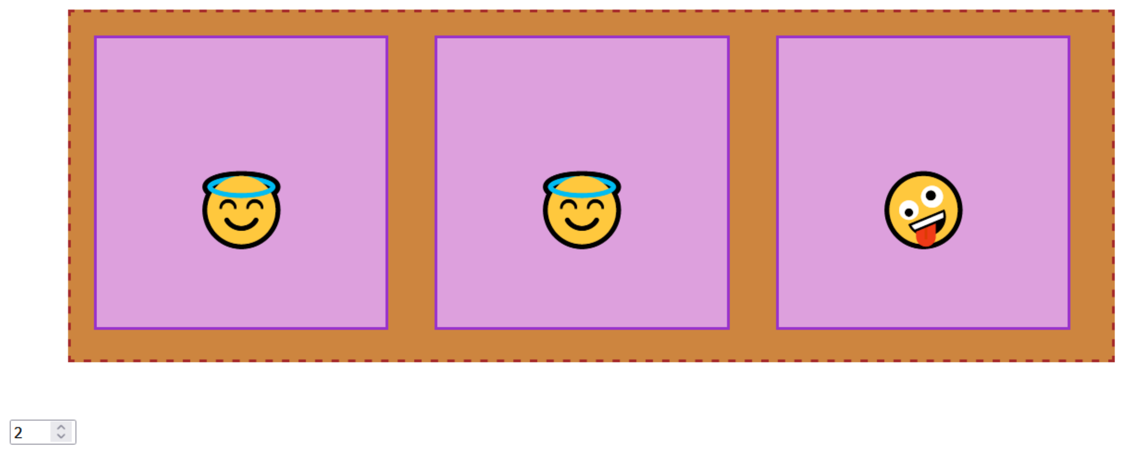
Sequelize is a promise-based Node.js ORM for Postgres, MySQL, MariaDB, SQLite and Microsoft SQL Server. It features solid transaction support, relations, eager and lazy loading, read replication and more.

Projects along the way:

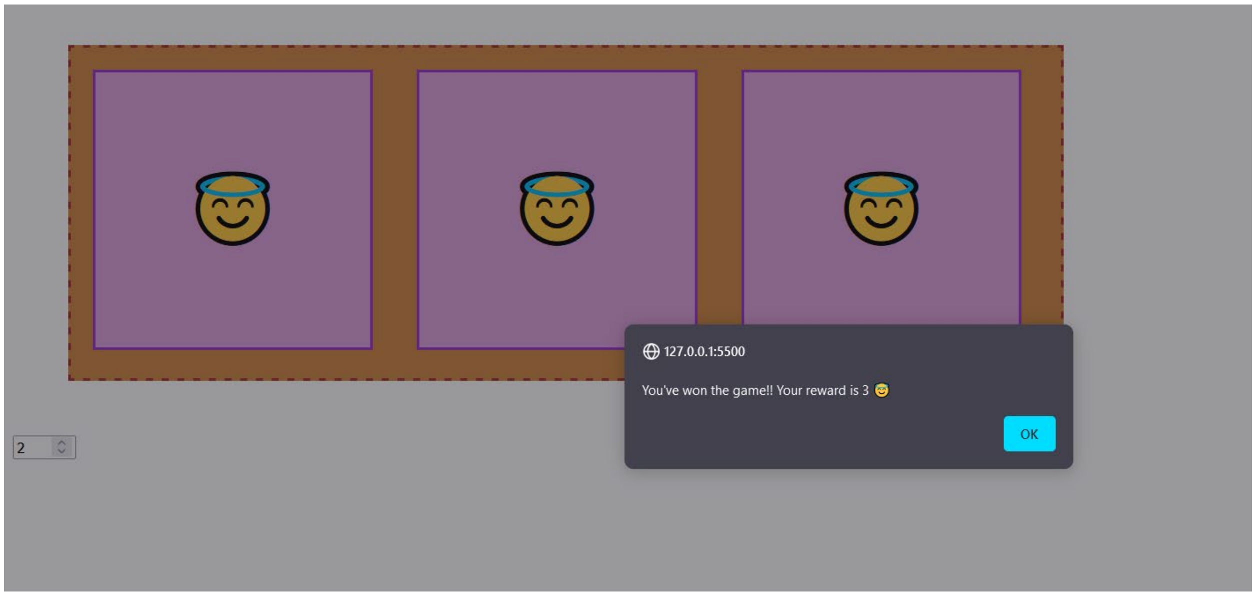
So, throughout the course, there were various small projects that I made to get myself familiarized with the concepts that I was learning along the way.

Slot machine:

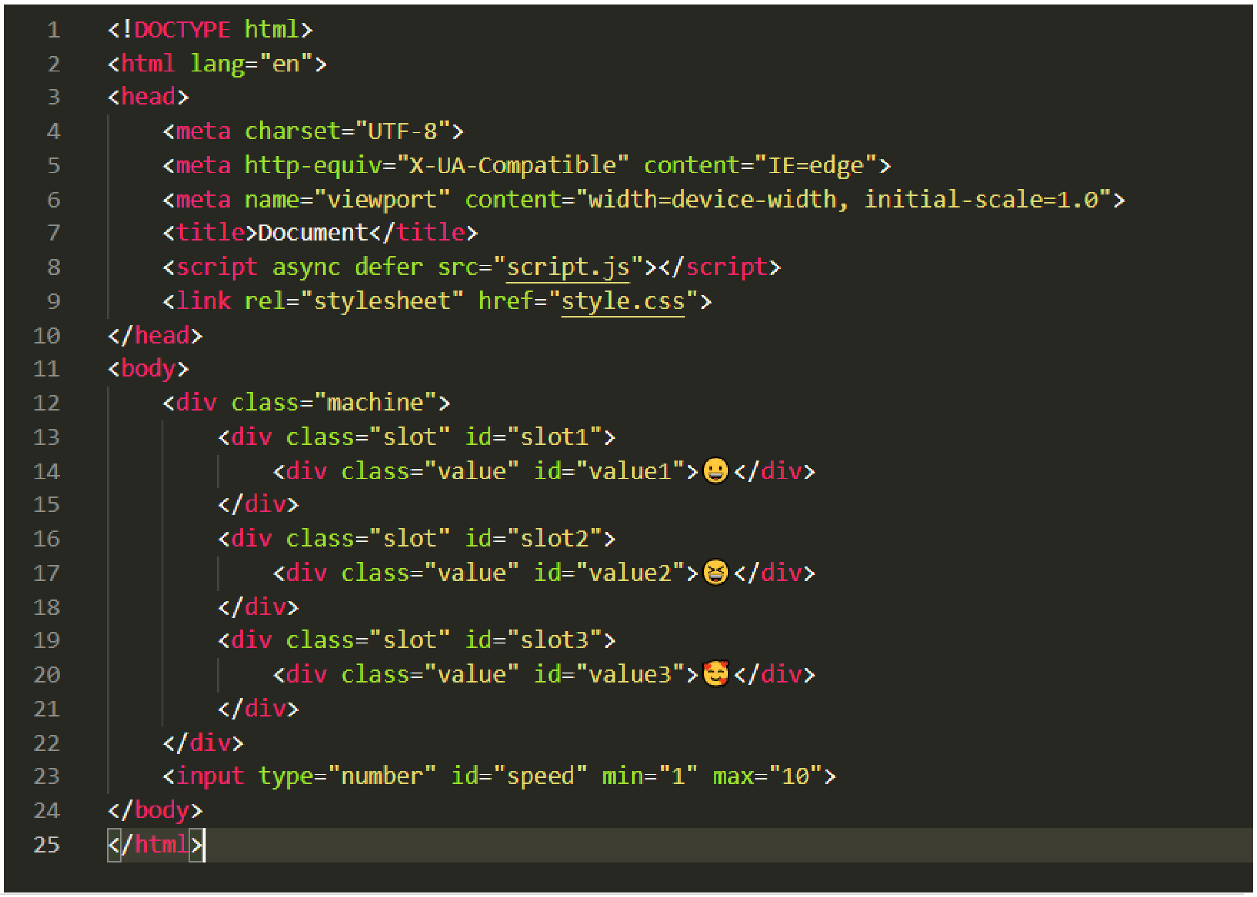
So, the first small project that I made was this slot machine. I used all 3 front end languages (HTML, CSS and JS) to make this. It's a basic slot machine that keeps rotating (There's a dial to set the speed at the bottom) and once all 3 slots show the same emoji, it prompts us that we've won the game



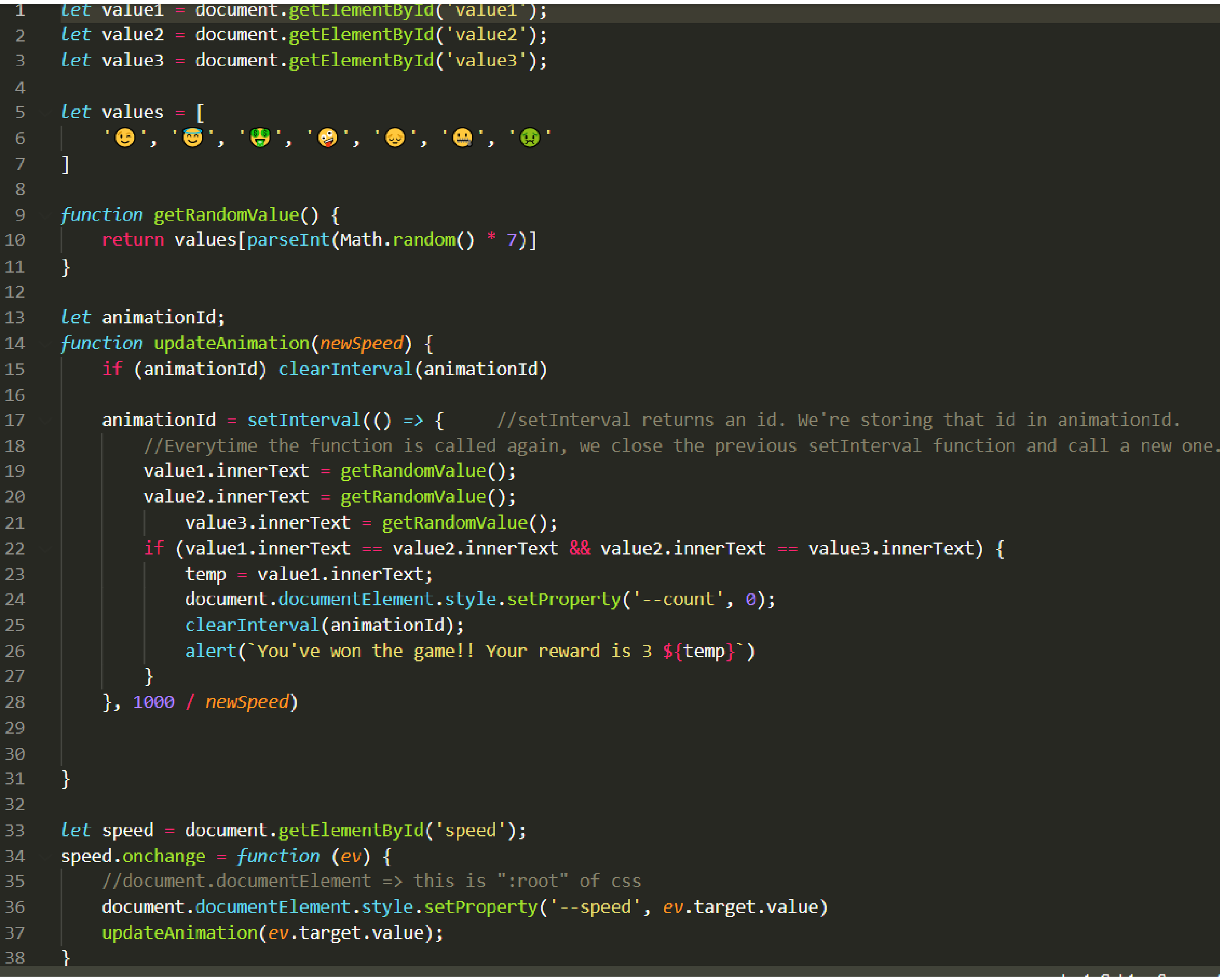
The slots keep spinning till all 3 show the same face.


```
127.0.0.1:5500
You've won the game!! Your reward is 3 😇
```

When the slots show the same emojis, it means that we've won the game and so the browsers shows an alert and it also shows us our reward.

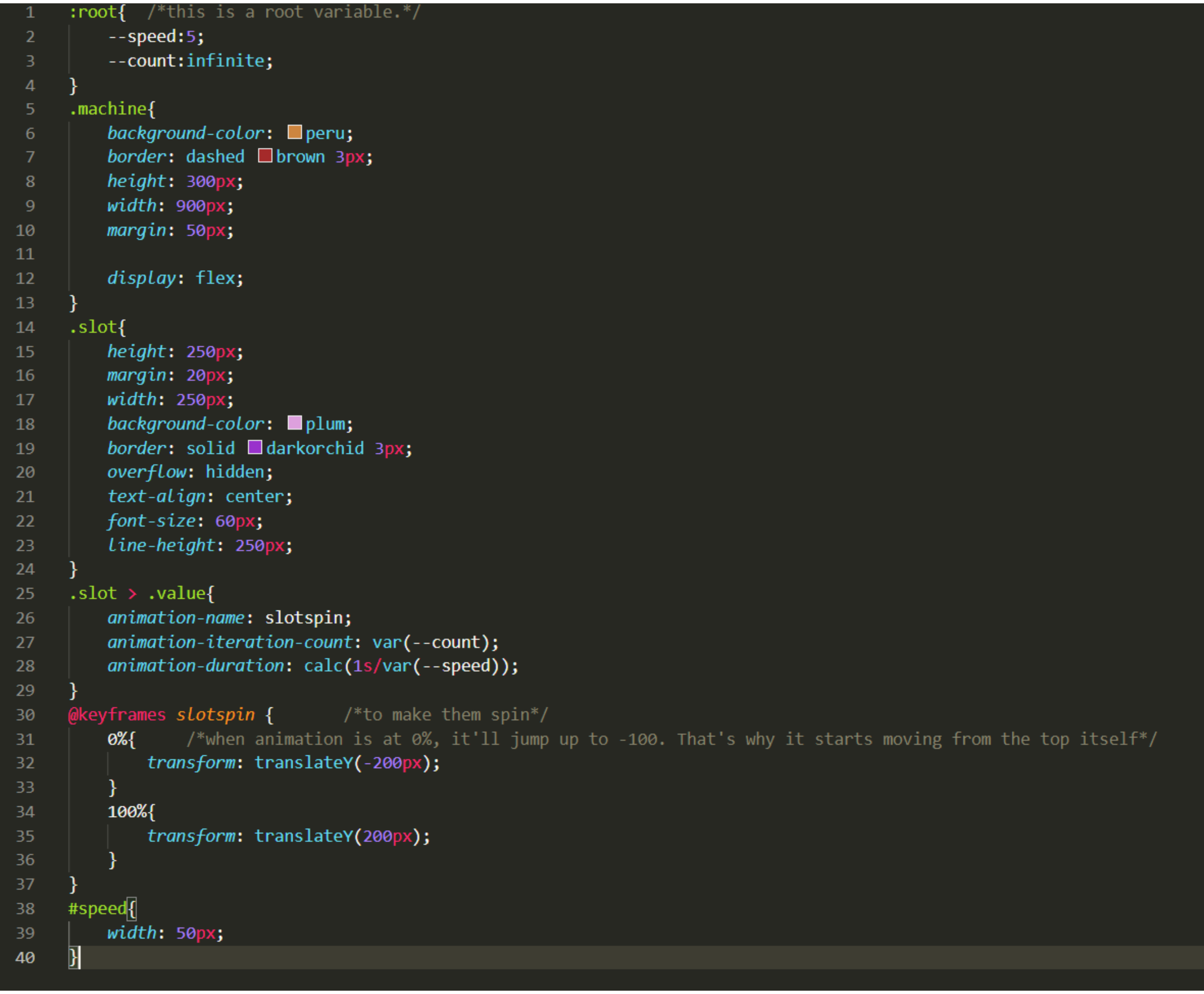

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <script async defer src="script.js"></script>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <div class="machine">
        <div class="slot" id="slot1">
            <div class="value" id="value1">😀</div>
        </div>
        <div class="slot" id="slot2">
            <div class="value" id="value2">😆</div>
        </div>
        <div class="slot" id="slot3">
            <div class="value" id="value3">🥰</div>
        </div>
    </div>
    <input type="number" id="speed" min="1" max="10">
</body>
</html>
```

JavaScript code:



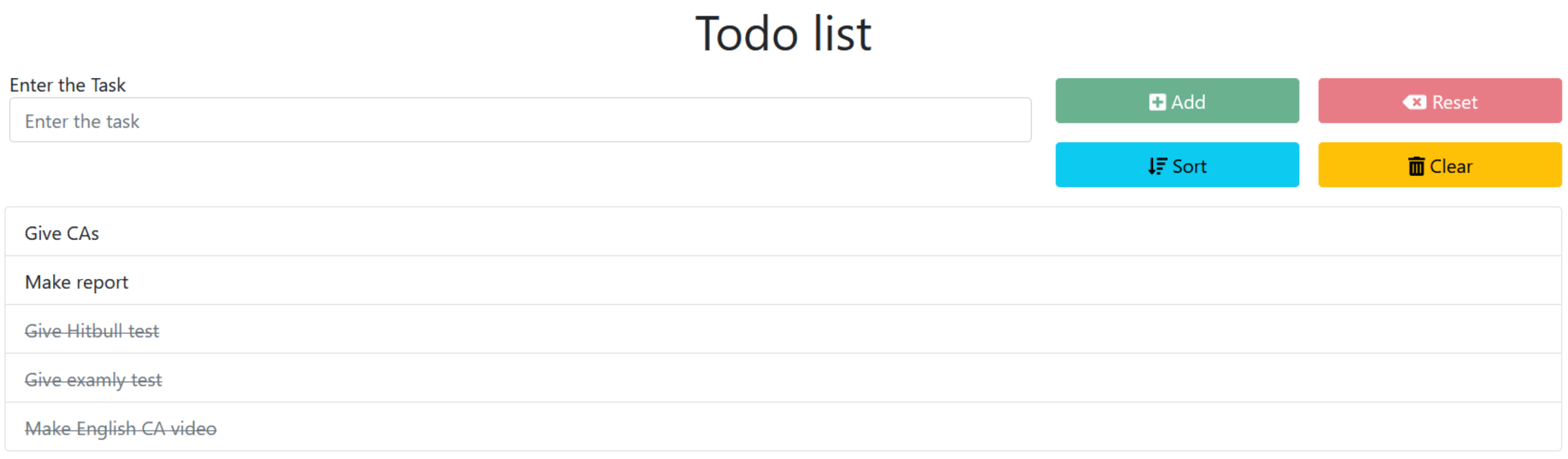
```javascript
let value1 = document.getElementById('value1');
let value2 = document.getElementById('value2');
let value3 = document.getElementById('value3');

let values = [
    '😉', '😇', '🤑', '🤪', '😔', '🤐', '🤢'
]

function getRandomValue() {
    return values[parseInt(Math.random() * 7)]
}

let animationId;
function updateAnimation(newSpeed) {
    if (animationId) clearInterval(animationId)

    animationId = setInterval(() => {     //setInterval returns an id. We're storing that id in animationId.
        //Everytime the function is called again, we close the previous setInterval function and call a new one.
        value1.innerText = getRandomValue();
        value2.innerText = getRandomValue();
            value3.innerText = getRandomValue();
        if (value1.innerText == value2.innerText && value2.innerText == value3.innerText) {
            temp = value1.innerText;
            document.documentElement.style.setProperty('--count', 0);
            clearInterval(animationId);
            alert(`You've won the game!! Your reward is 3 ${temp}`)
        }
    }, 1000 / newSpeed)


}

let speed = document.getElementById('speed');
speed.onchange = function (ev) {
    //document.documentElement => this is ":root" of css
    document.documentElement.style.setProperty('--speed', ev.target.value)
    updateAnimation(ev.target.value);
}
```

CSS code:



```css
:root{  /*this is a root variable.*/
    --speed:5;
    --count:infinite;
}
.machine{
    background-color: peru;
    border: dashed brown 3px;
    height: 300px;
    width: 900px;
    margin: 50px;

    display: flex;
}
.slot{
    height: 250px;
    margin: 20px;
    width: 250px;
    background-color: plum;
    border: solid darkorchid 3px;
    overflow: hidden;
    text-align: center;
    font-size: 60px;
    line-height: 250px;
}
.slot > .value{
    animation-name: slotspin;
    animation-iteration-count: var(--count);
    animation-duration: calc(1s/var(--speed));
}
@keyframes slotspin {       /*to make them spin*/
    0%{     /*when animation is at 0%, it'll jump up to -100. That's why it starts moving from the top itself*/
        transform: translateY(-200px);
    }
    100%{
        transform: translateY(200px);
    }
}
#speed{
    width: 50px;
}
```

To Do list:



Give CAs

Make report

~~Give Hitbull test~~

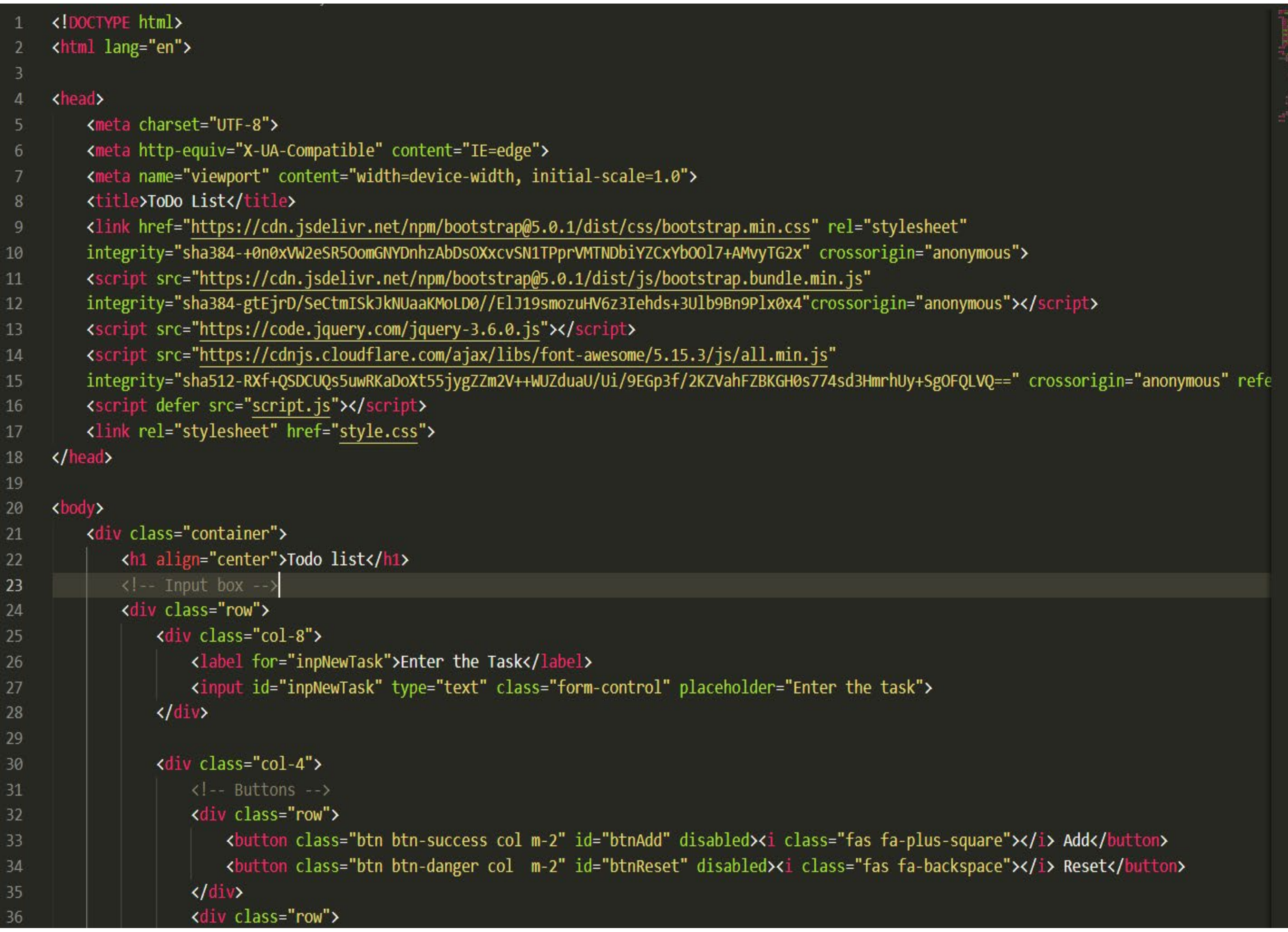
~~Give examly test~~

~~Make English CA video~~

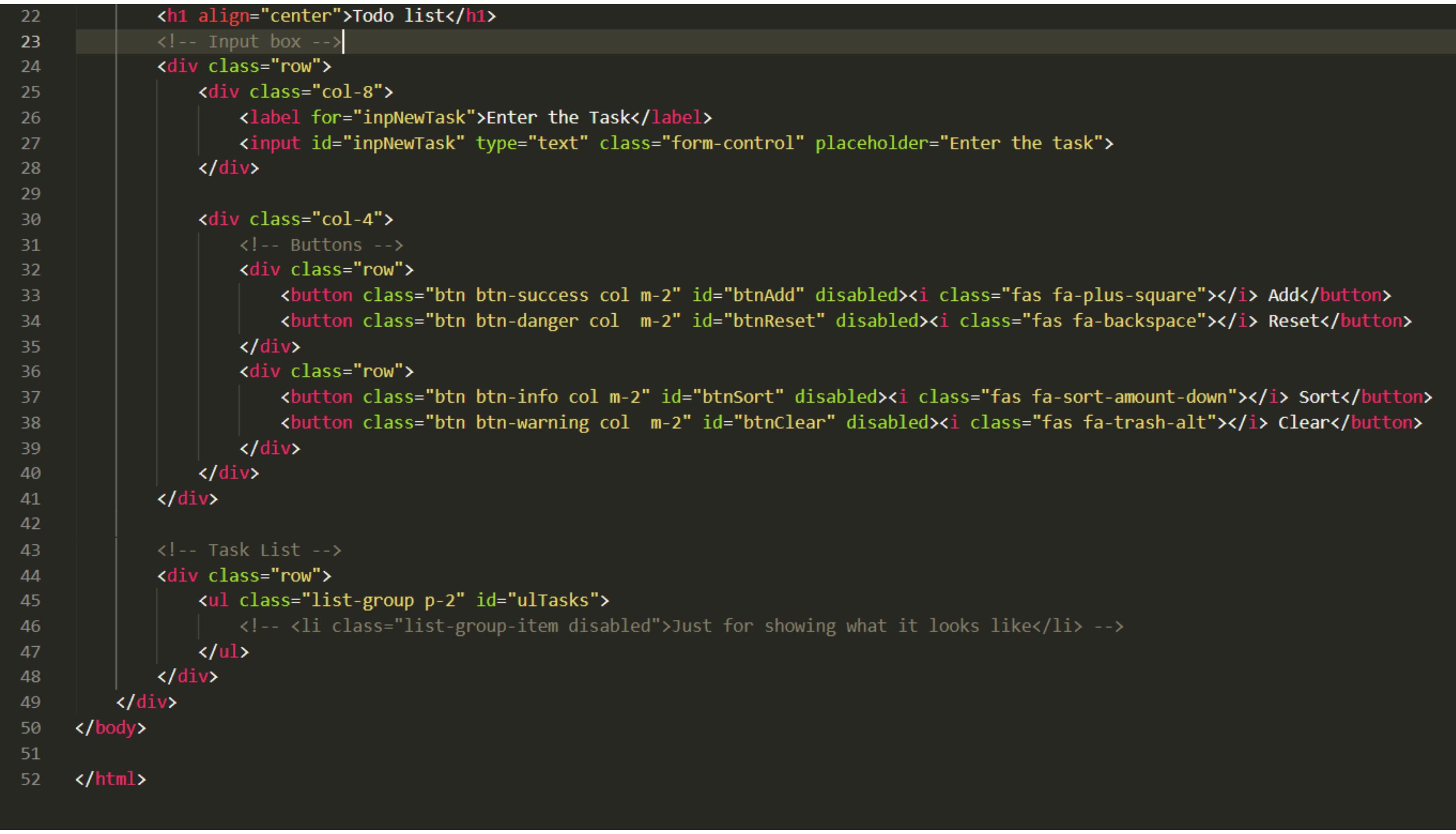
So this is a todo list that again utilizes HTML, CSS and Javascript. What happens here is, we can add tasks and mark them finished. If we click the sort button then that brings up all the remaining tasks and pushes down all the finished tasks. Pressing the clear button removes all the tasks that have been finished. We can mark a task finished/unfinished by simply clicking on it.

In this list, I also utilized a little bit of bootstrap to create the buttons and the input space and the list. Bootstrap is a library of CSS which is used for easier implementation of CSS and to utilize pre made classes for beautification.

HTML Code:



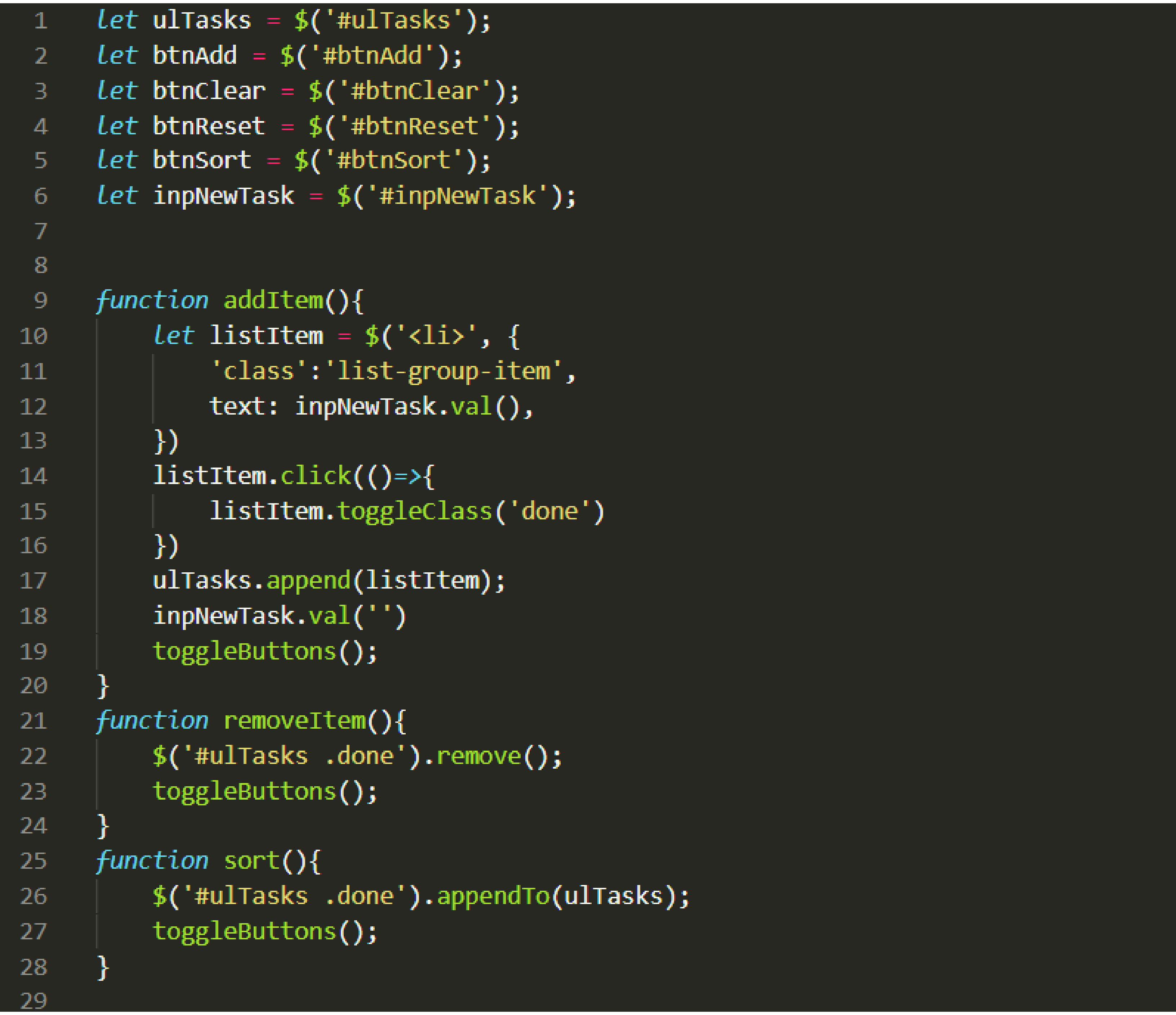
```html
<!DOCTYPE html>
<html lang="en">

<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">
    <title>ToDo List</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/bootstrap.min.css" rel="stylesheet"
    integrity="sha384-+0n0xVW2eSR5OomGNYDnhzAbDsOXxcvSN1TPprVMTNDbiYZCxYbOOl7+AMvyTG2x" crossorigin="anonymous">
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/js/bootstrap.bundle.min.js"
    integrity="sha384-gtEjrD/SeCtmISkJkNUaaKMoLD0//ElJ19smozuHV6z3Iehds+3Ulb9Bn9Plx0x4"crossorigin="anonymous"></script>
    <script src="https://code.jquery.com/jquery-3.6.0.js"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.3/js/all.min.js"
    integrity="sha512-RXf+QSDCUQs5uwRKaDoXt55jygZZm2V++WUZduaU/Ui/9EGp3f/2KZVahFZBKGH0s774sd3HmrhUy+SgOFQLVQ==" crossorigin="anonymous" refe
    <script defer src="script.js"></script>
    <link rel="stylesheet" href="style.css">
</head>

<body>
    <div class="container">
        <h1 align="center">Todo list</h1>
        <!-- Input box -->
        <div class="row">
            <div class="col-8">
                <label for="inpNewTask">Enter the Task</label>
                <input id="inpNewTask" type="text" class="form-control" placeholder="Enter the task">
            </div>

            <div class="col-4">
                <!-- Buttons -->
                <div class="row">
                    <button class="btn btn-success col m-2" id="btnAdd" disabled><i class="fas fa-plus-square"></i> Add</button>
                    <button class="btn btn-danger col  m-2" id="btnReset" disabled><i class="fas fa-backspace"></i> Reset</button>
                </div>
                <div class="row">
```



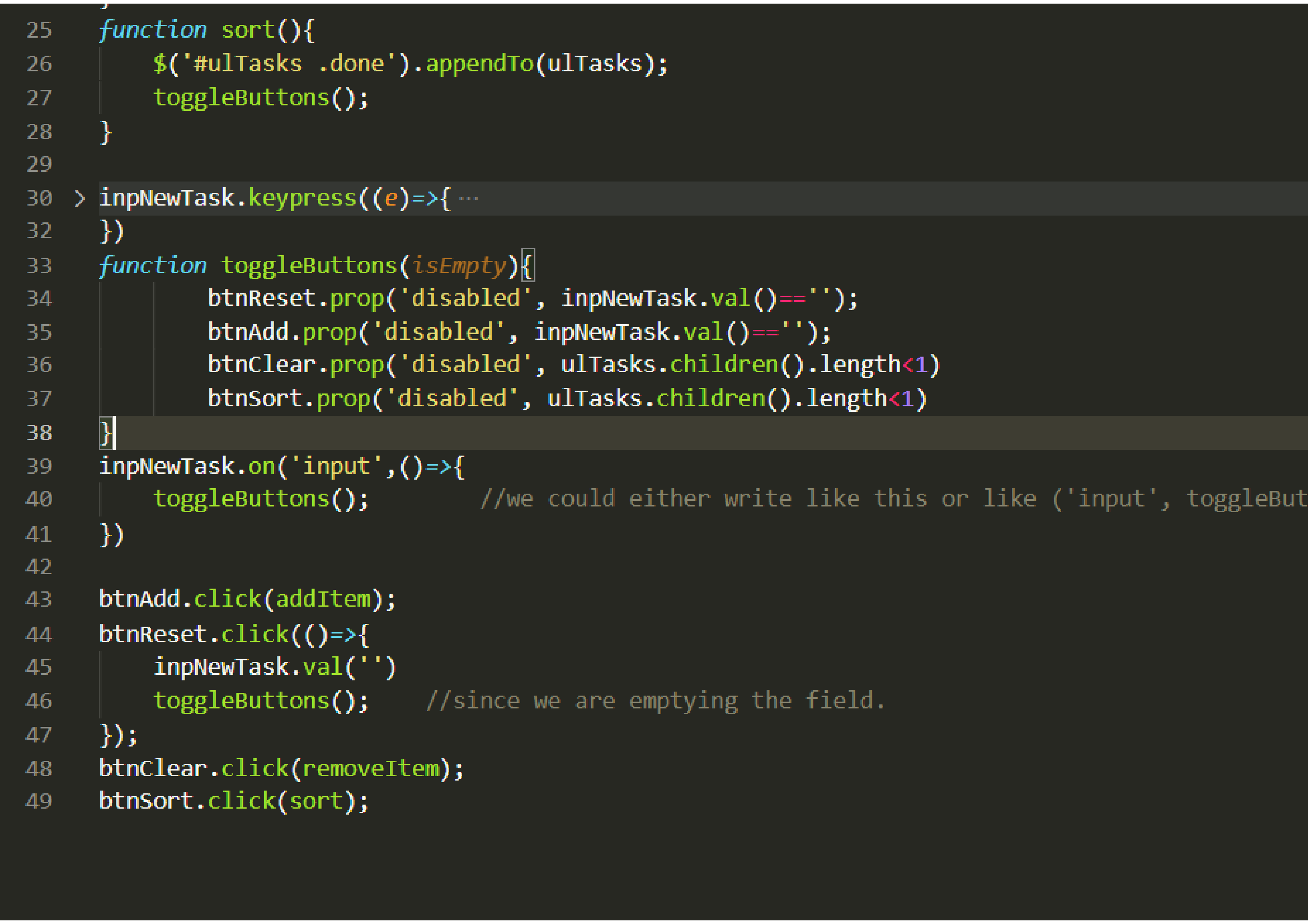
```
        <h1 align="center">Todo list</h1>
        <!-- Input box -->
        <div class="row">
            <div class="col-8">
                <label for="inpNewTask">Enter the Task</label>
                <input id="inpNewTask" type="text" class="form-control" placeholder="Enter the task">
            </div>

            <div class="col-4">
                <!-- Buttons -->
                <div class="row">
                    <button class="btn btn-success col m-2" id="btnAdd" disabled><i class="fas fa-plus-square"></i> Add</button>
                    <button class="btn btn-danger col  m-2" id="btnReset" disabled><i class="fas fa-backspace"></i> Reset</button>
                </div>
                <div class="row">
                    <button class="btn btn-info col m-2" id="btnSort" disabled><i class="fas fa-sort-amount-down"></i> Sort</button>
                    <button class="btn btn-warning col  m-2" id="btnClear" disabled><i class="fas fa-trash-alt"></i> Clear</button>
                </div>
            </div>
        </div>

        <!-- Task List -->
        <div class="row">
            <ul class="list-group p-2" id="ulTasks">
                <!-- <li class="list-group-item disabled">Just for showing what it looks like</li> -->
            </ul>
        </div>
    </div>
</body>

</html>
```

This was the first time when I used bootstrap and honestly speaking, I'm not that familiar with it and I had some issues here and there while creating this code. But I learnt that bootstrap is extremely useful for frontend as there are many classes in it and they can be used very extensively to not only make your front end beautiful but to ease your work as a developer.

JS code:



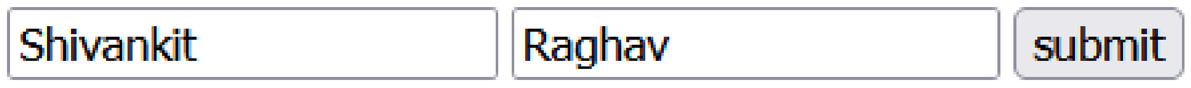
```
let ulTasks = $('#ulTasks');
let btnAdd = $('#btnAdd');
let btnClear = $('#btnClear');
let btnReset = $('#btnReset');
let btnSort = $('#btnSort');
let inpNewTask = $('#inpNewTask');


function addItem(){
    let listItem = $('<li>', {
        'class':'list-group-item',
        text: inpNewTask.val(),
    })
    listItem.click(()=>{
        listItem.toggleClass('done')
    })
    ulTasks.append(listItem);
    inpNewTask.val('')
    toggleButtons();
}
function removeItem(){
    $('#ulTasks .done').remove();
    toggleButtons();
}
function sort(){
    $('#ulTasks .done').appendTo(ulTasks);
    toggleButtons();
}

```


```
function sort(){
    $('#ulTasks .done').appendTo(ulTasks);
    toggleButtons();
}

inpNewTask.keypress((e)=>{ ···
})
function toggleButtons(isEmpty){
        btnReset.prop('disabled', inpNewTask.val()=='');
        btnAdd.prop('disabled', inpNewTask.val()=='');
        btnClear.prop('disabled', ulTasks.children().length<1)
        btnSort.prop('disabled', ulTasks.children().length<1)
}
inpNewTask.on('input',()=>{
    toggleButtons();        //we could either write like this or like ('input', toggleBut
})

btnAdd.click(addItem);
btnReset.click(()=>{
    inpNewTask.val('')
    toggleButtons();    //since we are emptying the field.
});
btnClear.click(removeItem);
btnSort.click(sort);
```

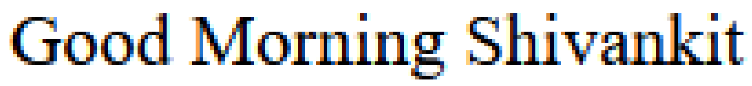
This small project was actually pretty fun to make as I had a lot of fun with using javascript.

Greeter using Express:

Express was a little bit difficult for me to understand and I hope to learn more about it in my classes. But I still tried something small and created this greeter in which you type your name and it greets you by saying good morning.

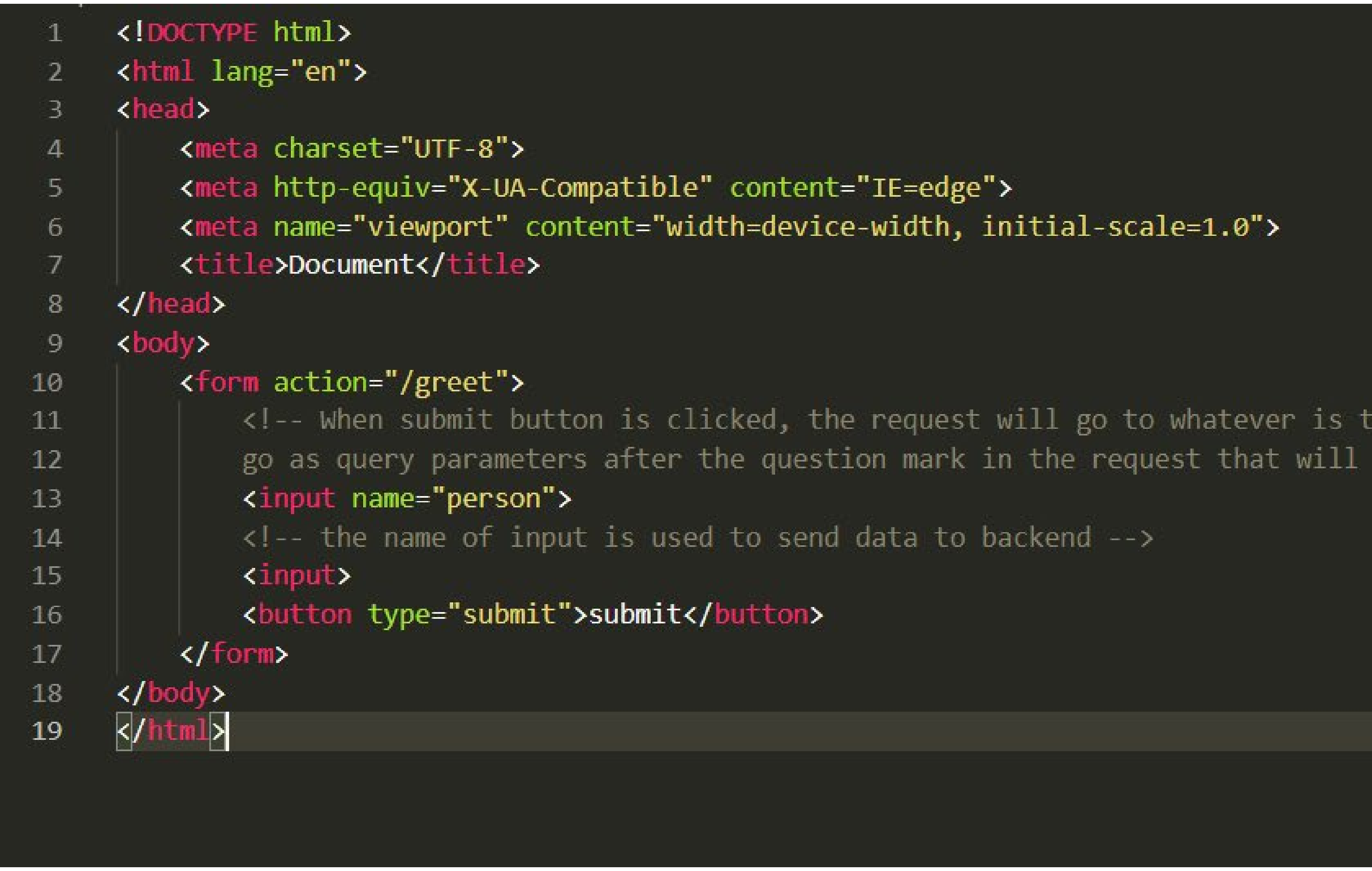


This input box opens up in your browser. Once you type your name and click send, this input goes to the backend using a ‘get’ request. And in response of that, the server greets you on the page like this:



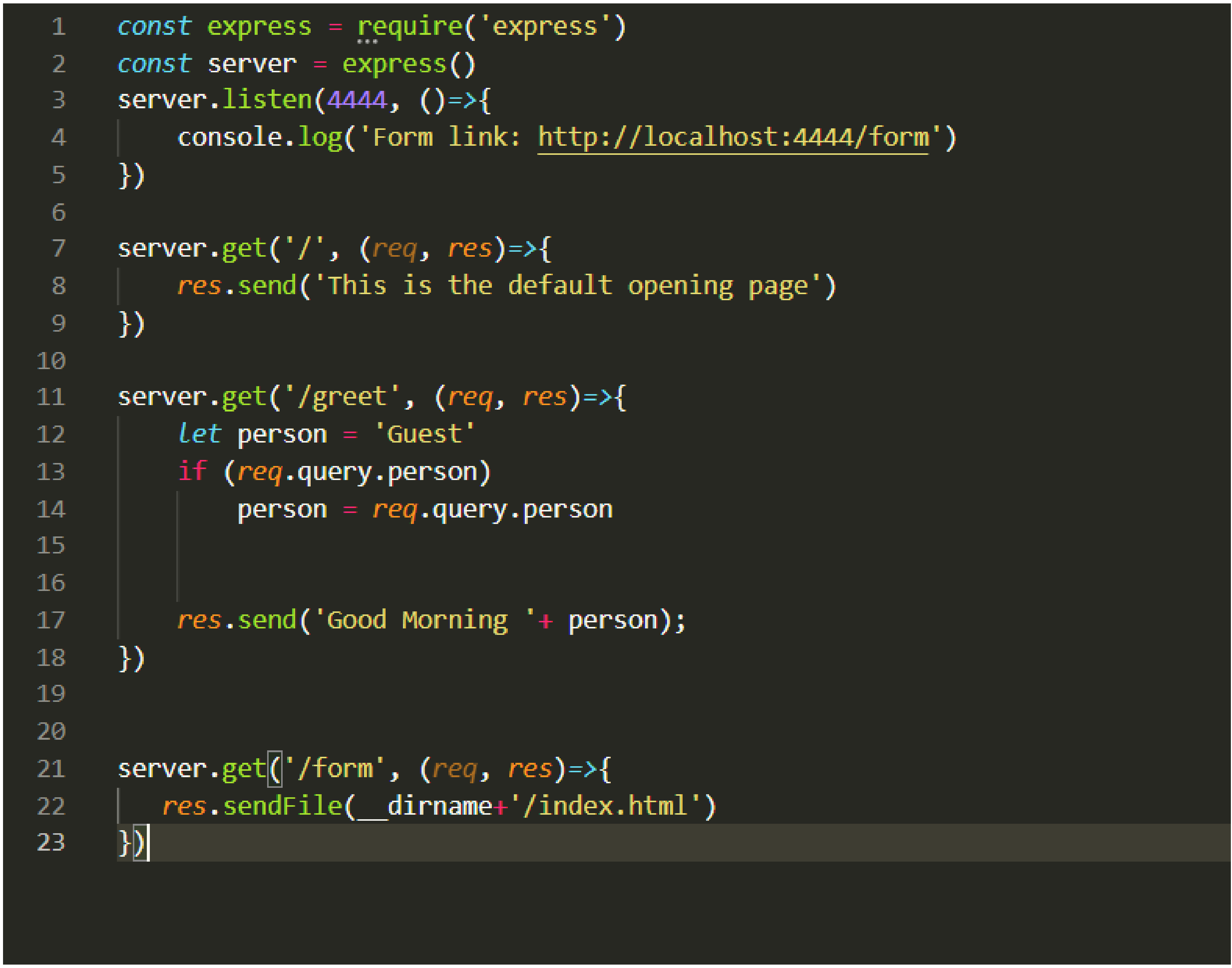
Good Morning Shivankit

For this, I simply used a little bit of HTML and a little bit of Javascript along with Express.



```html
<!DOCTYPE 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">
    <title>Document</title>
</head>
<body>
    <form action="/greet">
        <!-- When submit button is clicked, the request will go to whatever is t
        go as query parameters after the question mark in the request that will
        <input name="person">
        <!-- the name of input is used to send data to backend -->
        <input>
        <button type="submit">submit</button>
    </form>
</body>
</html>
```

When the submit button is clicked, the ‘action’ will send a get request to ‘/greet’ path and that get request will contain the value of input box with id ‘person’. Then the server will use that information to print the name on the webpage.



```javascript
const express = require('express')
const server = express()
server.listen(4444, ()=>{
    console.log('Form link: http://localhost:4444/form')
})

server.get('/', (req, res)=>{
    res.send('This is the default opening page')
})

server.get('/greet', (req, res)=>{
    let person = 'Guest'
    if (req.query.person)
        person = req.query.person


    res.send('Good Morning '+ person);
})


server.get('/form', (req, res)=>{
   res.sendFile(__dirname+'/index.html')
})
```

Here, the server starts at port 4444. If we don't send any data from the front end then the default value of 'person' will be initialized as 'guest' and it'll print "Good Morning Guest"

Reason to choose this technology

Web design and development is the challenging field and it's rewarding too. In this field, we need to implement our imagination and creativity skills to develop attractive website based on the client business and their preference. Probably most of the web developers chose web design field because they are very passionate in doing design and develop. Both the skills and personal experience of the designer will help developer to develop stunning web applications.

- ☐ In web development, there is huge career opportunity for skilled and talented professionals. The factors like job role, job location and experience will determine the salary package of the web designers and developers.
- ☐ Web designer can create attractive website using various tool and resources to satisfy the client requirements. We need to focus on various creative areas and tools to develop a website. We need to also implement our own imagination skills to be a skilled developer. Though web design and development is a challenging career, it can be learned with ease.
- ☐ We can also work as a freelance developer. Website is most effective and efficient tool to promote our business targeting the global customer base. In general, website with attractive interface and good quality content will perform better compared to mere static website. We can include more multimedia

content to improve website appearance and navigation.

- Most promising and profitable, web development is most trending field of interest. Nowadays, most of the work is done online, which requires a trustworthy website, which requires a web developer with vast knowledge and skills but still well experienced and knowledgeable web developers can make a great profit.

It’s estimated that there are over 1.7 billion websites present on the world wide web. And that number is so big because with the increasing technology, more and more people are getting access to internet and more and more people are getting interested in websites and are learning to make websites even at home. Every small and large business has a website now-a-days and that simply means that the need of web developers is ever increasing.

Learning Outcomes

After completion of extensive two months summer training on the course of Web Development, I had a brief and thorough knowledge of various terms and technologies related to Web Development. It in itself is a very wide subject with numerous opportunities to grab, to excel in the field of web developing all we need is to explore our imagination and think outside the box. As we all know nowadays everything is going online, from the simplest thing to the most complex jobs in the world, everything takes place on internet and for those to happen, we need a steady platform, which is provided by a reliant web site.

A website is not limited; we can extend its use up to no limit and reach great outcomes. I, through the course of two months of web development, was able to learn some of the technologies which played a vital role in web development.

The very first technology or in simpler words language I learnt was HTML, which extends to Hyper Text Markup Language, this is the most basic technology used for the development of web pages. We start from the most basic but most important and reach to the level of perfection, so in web development we need to start from HTML, and steadily need to reach to perfection.

HTML is used for the development of web pages and web applications, from very basic web pages containing just a form or table to the most advanced web pages which includes great amount of animation and various other parameters can be created using HTML. It can be paired up with other languages in order to create more efficient web pages. In conclusion, I would say that through the course I was able to grab efficient and useful knowledge which would definitely help me in future, in this era of internet.

Now HTML can be easily paired up with several other technologies for creation of great web pages. Out of many available, I was able to get knowledge of CSS, JavaScript, and PHP. All these languages play great amount of role for the formation of a web page.

First, I would talk about CSS. Now CSS expands to Cascading Style Sheet, this language cannot be used alone as it is necessary for it to be paired up with HTML. This language is not used for creating any webpage but for providing various attributes to it, in simpler words we can say, to provide styling to the web page. CSS can be used to from the very simple, like changing font of text, to achieve complex jobs, like defining transitions for slideshow on a web page, and various other parameters. No one would like to look at static web pages with very less or no animations and with less amount of styling. So, to make a page look more attractive and effective CSS is used. When we use web application or website or any interface, we don't want it to look unattractive and unappealing, we always want it to be catchy, therefore to achieve such we need to use CSS. HTML paired up with CSS can result in the formation of attractive and appealing web pages. I was able to learn CSS up to some extent during training.

Next on our list we have JavaScript, this language is a bit complex from the two discussed above. Sometimes we need to program our website, we cannot introduce several functions with HTML and CSS, but are possible with the help of JavaScript, it enables user to define various functions which enhances usability. JavaScript, termed as JS, is a programming language, just like C, C++ and most certainly Java.

It improves the functioning of webpage, as we can do great amount of programming for our website. JS is also used in combination with HTML. Alone JS cannot withstand responsibility of designing an entire web page, it needs to be paired up with JavaScript.

JS is programmed under script tag in HTML. Overall, if we want our website to be more effective then HTML, CSS and JS are combined.

HTML, CSS and JS do not have ability to store into databases, they can only be used for front end, for what the user sees. And this is where Express comes into play. During the course, I learnt how to create basic web servers using express and got excited for backend development. We also need a Database to use for websites and so we used a little bit of mySQL and then after that, relied on Sequelize for easier access. We also at one small point used MongoDB which is a NoSQL database. NoSQL databases are actually the once that are most often used and most often relatable with real life problems. All in all, the backend part of the course was not as detailed as the front end but it was good enough to increase my interest in Backend and I hope to learn more about it in the future.

So, in conclusion I would say that I was able to grab knowledge about these important web technologies which are important for a front-end web developer and a backend developer.

After the completion of training, I was able to create a simple basic yet working website. The most important outcome of the training was that now I am having a sound knowledge of various important technologies for web development and I have strong belief that these will always help me in future.