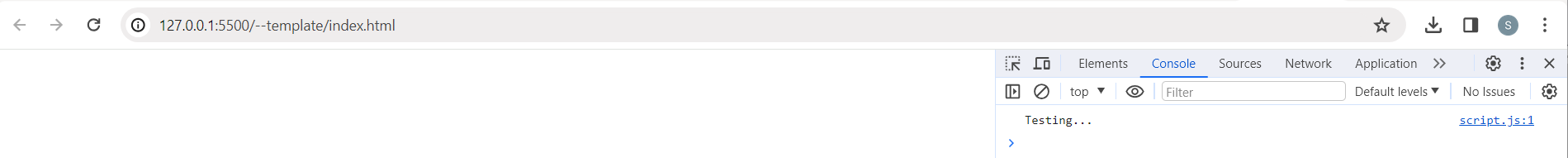
Note: Ctrl+/ to get the comment line

1.To check that JS is linked with html, we are writing to the console window in script.js file

console.info('Testing...');



Note: To host the files to github

a)Copy the file to github

PS C:\Users\HP\Desktop\JScourse> **git add .**

PS C:\Users\HP\Desktop\JScourse> **git commit -m "added html-css-js file"**

[main 09daa27] added html-css-js file

3 files changed, 34 insertions(+)

create mode 100644 --template/index.html

create mode 100644 --template/script.js

create mode 100644 --template/style.css

PS C:\Users\HP\Desktop\JScourse> **git push -u origin main**

info: please complete authentication in your browser...

Enumerating objects: 7, done.

Counting objects: 100% (7/7), done.

Delta compression using up to 8 threads

Compressing objects: 100% (5/5), done.

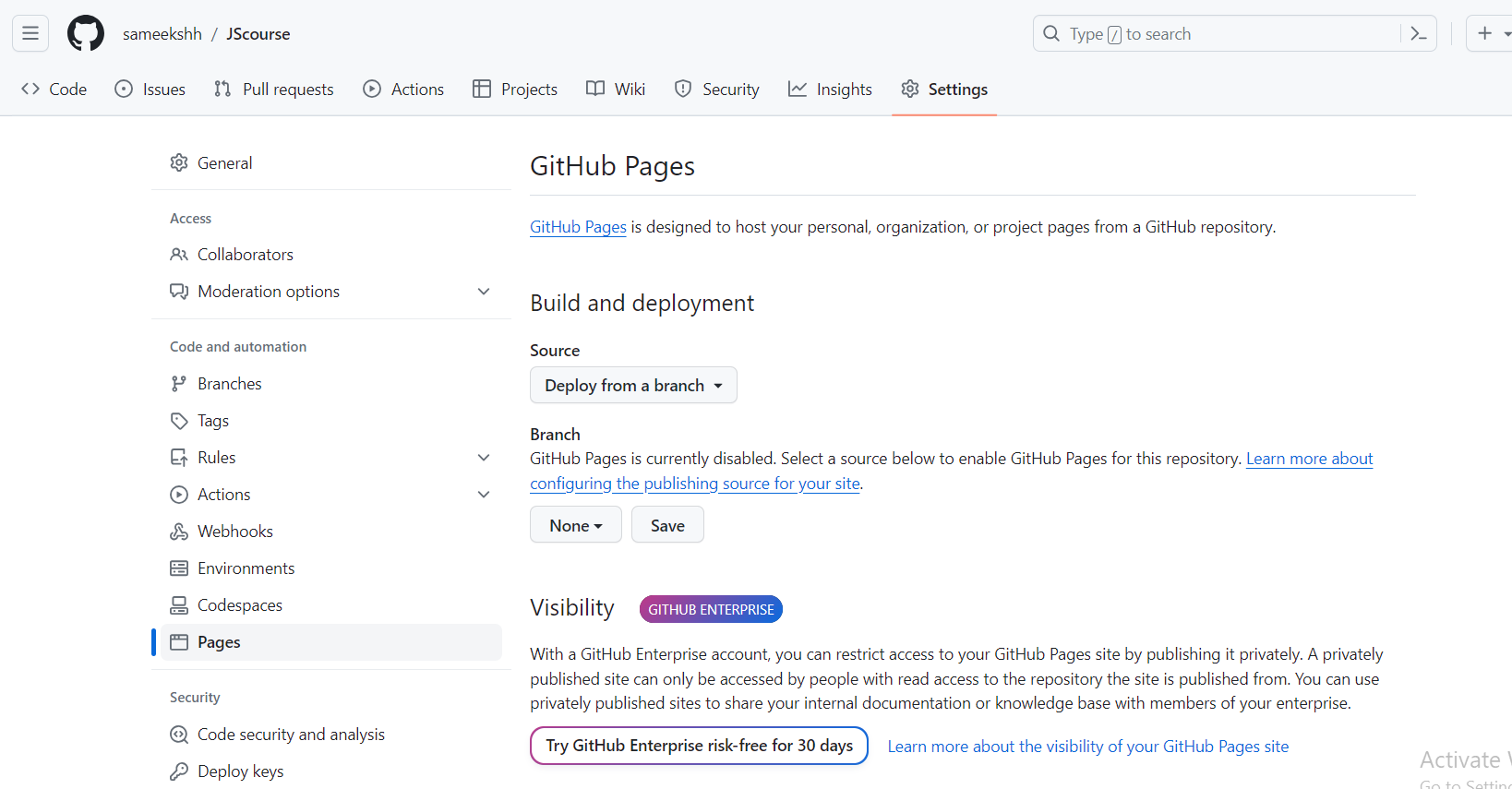
Writing objects: 100% (6/6), 952 bytes | 952.00 KiB/s, done.

Total 6 (delta 0), reused 0 (delta 0), pack-reused 0

To https://github.com/sameekshh/JScourse.git

eeafa96..09daa27 main -> main

branch 'main' set up to track 'origin/main'.

b)Go to settings in github:

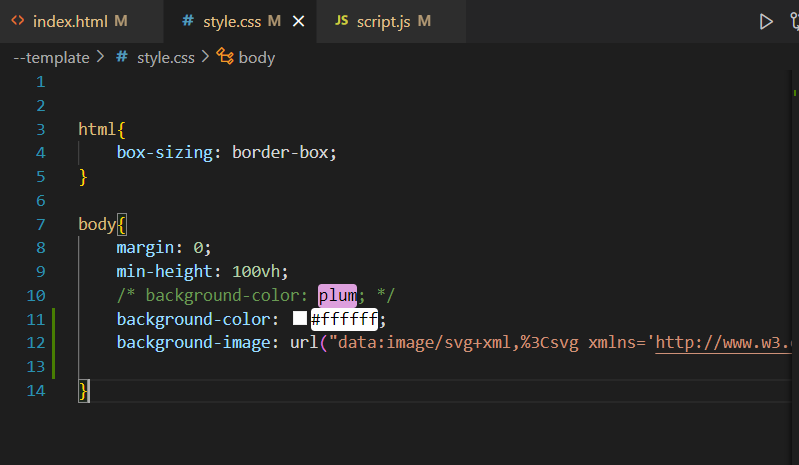
Note: Github page create is <https://sameekshh.github.io/JScourse/>

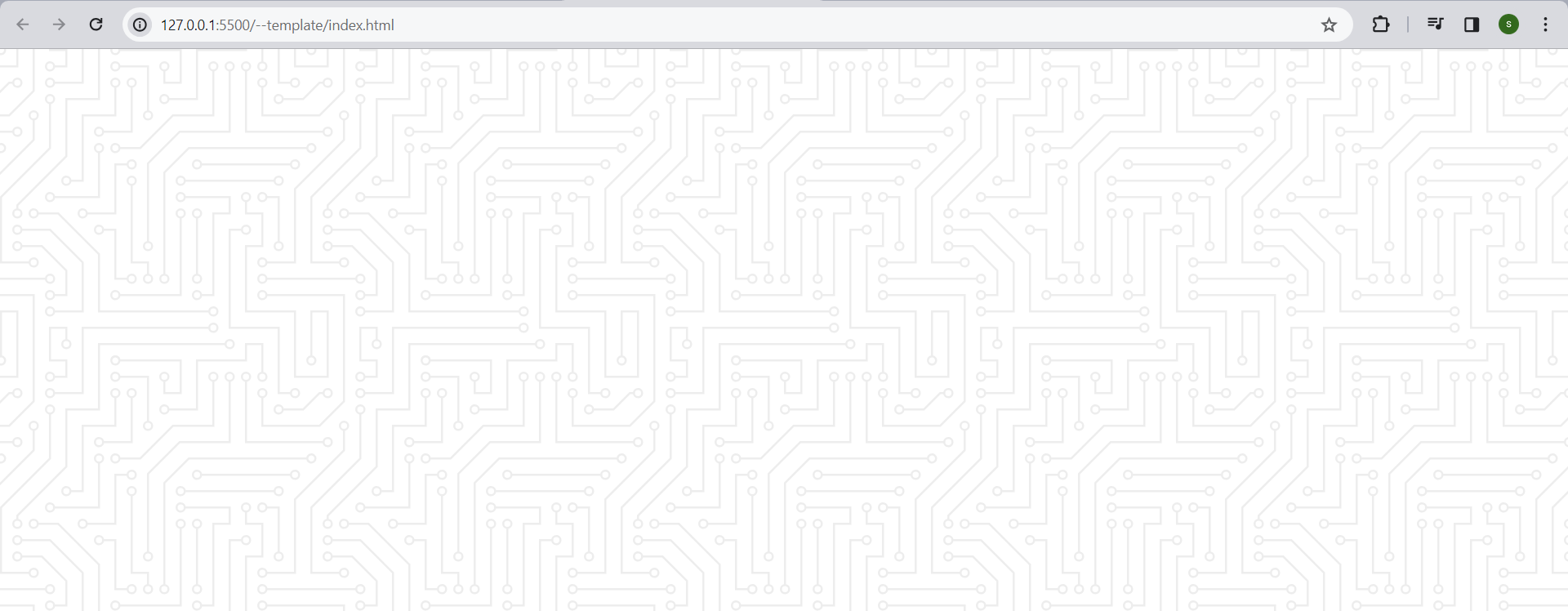
**Quote Generator**

1,Website to choose background pattern: (This is a SVG[scalable Vector Graphics] )

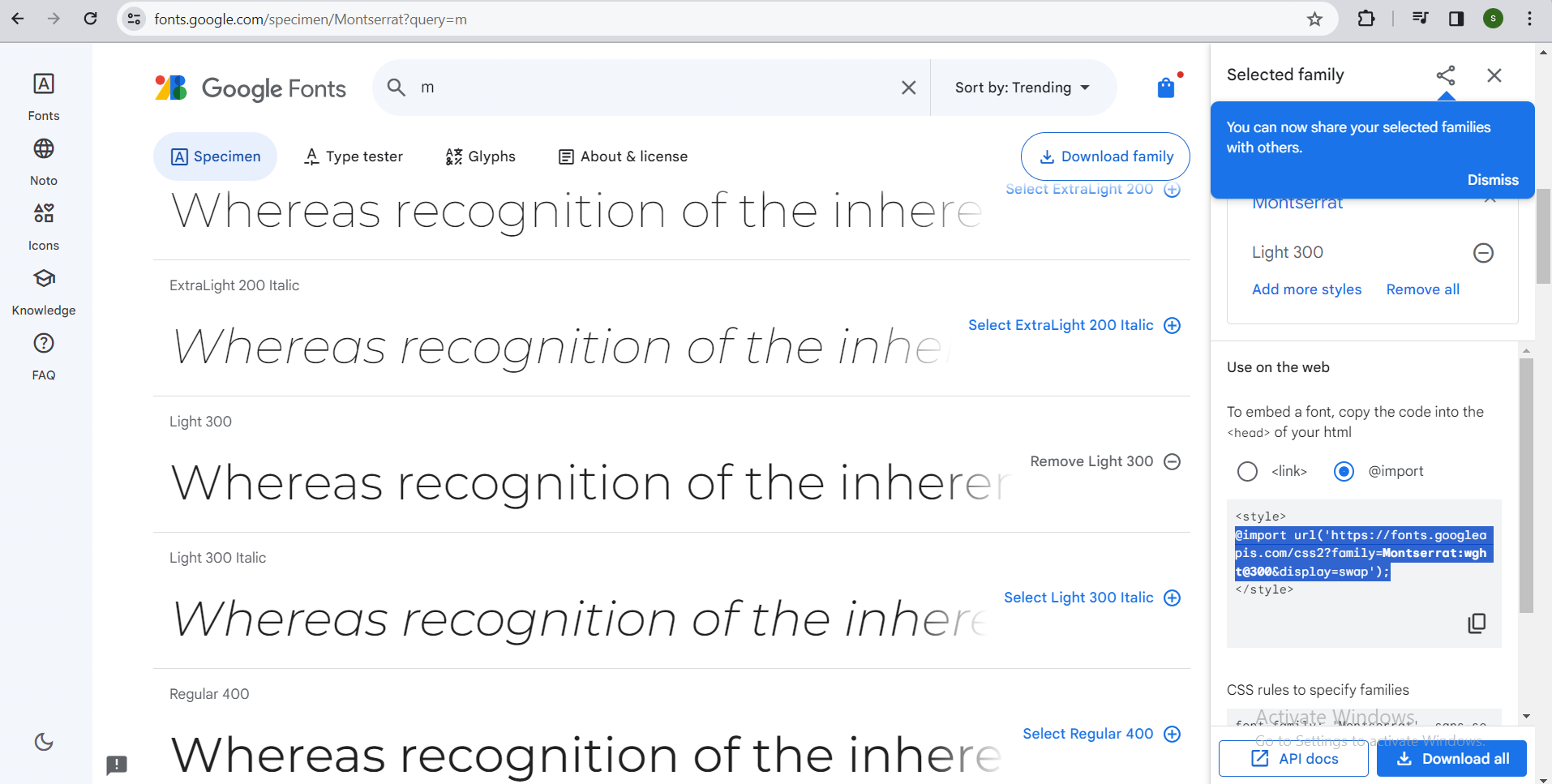
<https://heropatterns.com/>

It gives the CSS content, that needs to added in the body





The background is dimmed out so it is not visible properly.

2.Go to the website : google fonts. And copy the import portion only

Add this import in the top of CSS file.

Note:The display: flex; activates the flexbox layout, and align-items: center; and justify-content: center; work together to ensure that the child elements are centered both vertically and horizontally.

@import url('https://fonts.googleapis.com/css2?family=Montserrat:wght@300&display=swap');

html{

box-sizing: border-box;

}

body{

margin: 0;

min-height: 100vh;

/\* background-color: plum; \*/

background-color: #ffffff;

background-image: url("data:image/svg+xml,%3Csvg xmlns='http://www.w3.org/2000/svg'...");

/\* Give some formatting to the fonts \*/

color: #000;

font-family: Montserrat, sans-serif;

font-weight: 700;

text-align: center;

display: flex;

align-items: center; /\*vertical alignment\*/

justify-content: center; /\*horizontal alignment\*/

}

3.We can use emmet abbreviation in VScode, to get things done faster

To add a class with ID: div.my-class#my-id

This gives the following: <div class="my-class" id="my-id"></div>

Creating a quotes and author container:

<div class="quote-container" id="quote-container">

<!-- quote -->

<div class="quote-text">

<i class="fas fa-quote-left"></i>

<span id="quote">What you are is what you have been.What you'll be is what you do now.</span>

</div>

</div>

<!-- Author -->

<div class="quote-autor">

<span id="author">Buddha</span>

</div>

Adding in two buttons in:

<!-- Buttons -->

<div class="button-container">

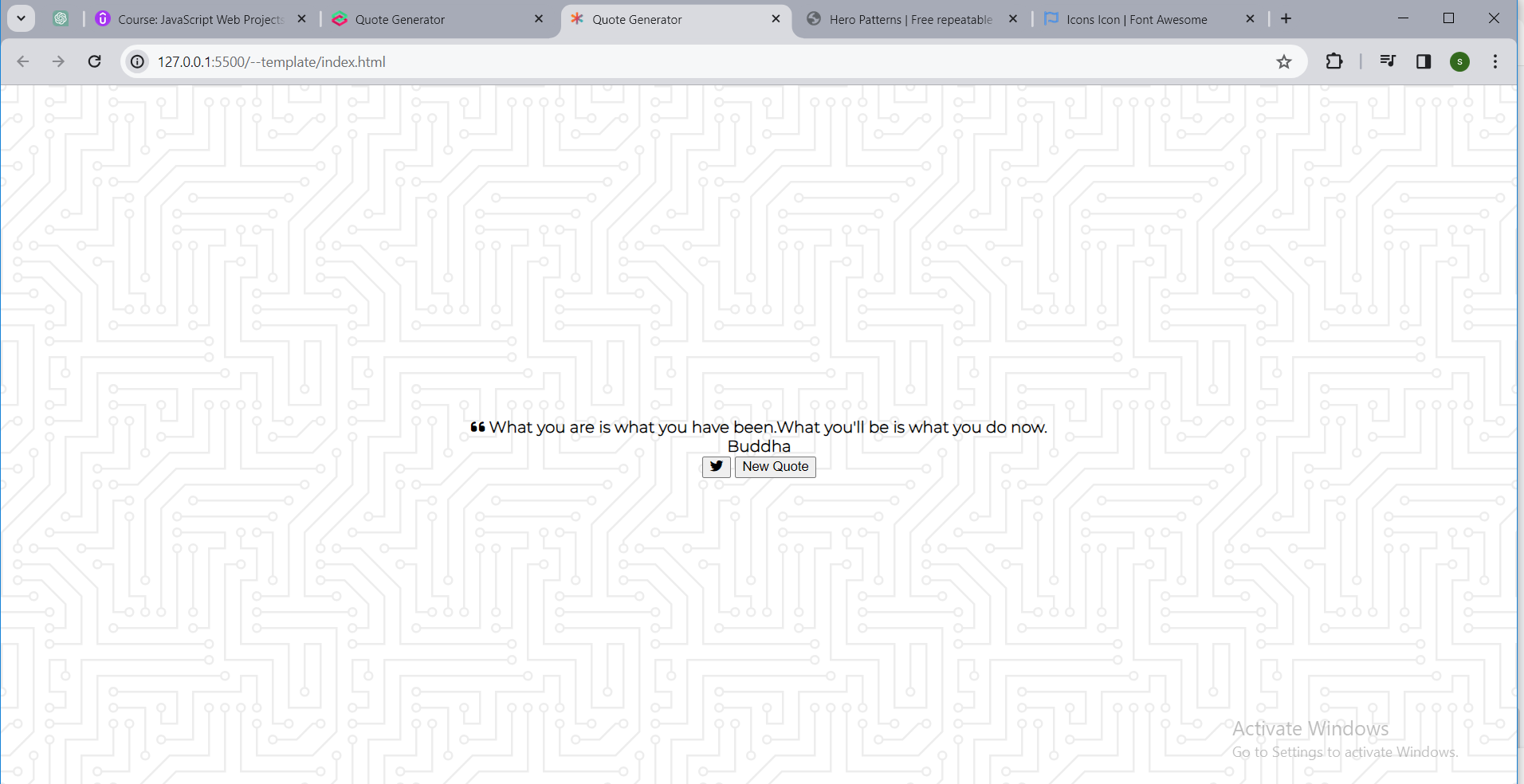
<button class="twitter-button" id="twitter" title="Tweet This!">

<i class="fa-brands fa-twitter"></i>

</button>

<button id="new-quote">New Quote</button>

</div>



4.Add the style for box container:

.quote-container {

width: auto;

max-width: 900px;

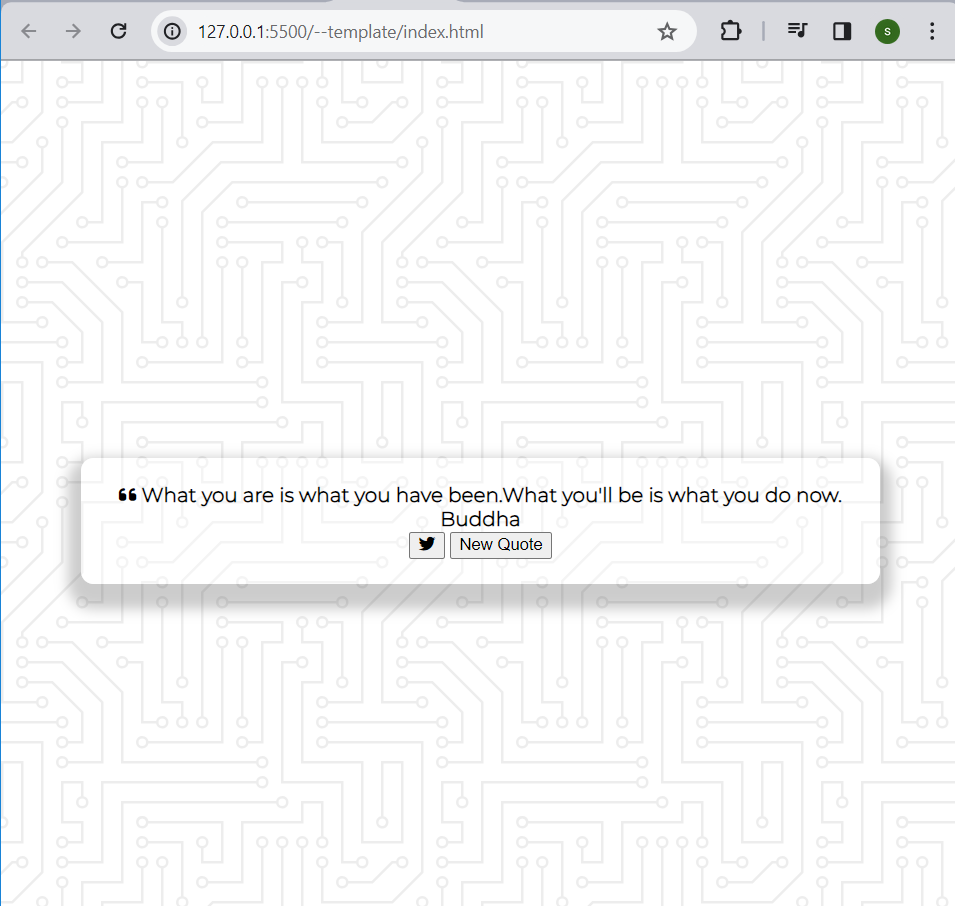
padding: 20px 30px; /\*top&bottom 20px\*/

border-radius: 10px; /\*This gives container a rounded corner\*/

background-color: rgba(255, 255, 255, 0.4);

box-shadow: 0 10px 10px 10px rgba(0, 0, 0, 0.2);

}



5.Stylization of the quote and author fonts. Here one option is provided when quote is too long.

.quote-text {

font-size: 2.75rem;/\*rem=16px\*/

}

.long-quote{ /\*This is dynamically turned on-off using JS\*/

font-size: 2rem;

}

.fa-quote-left {

font-size: 4rem;

}

.quote-author {

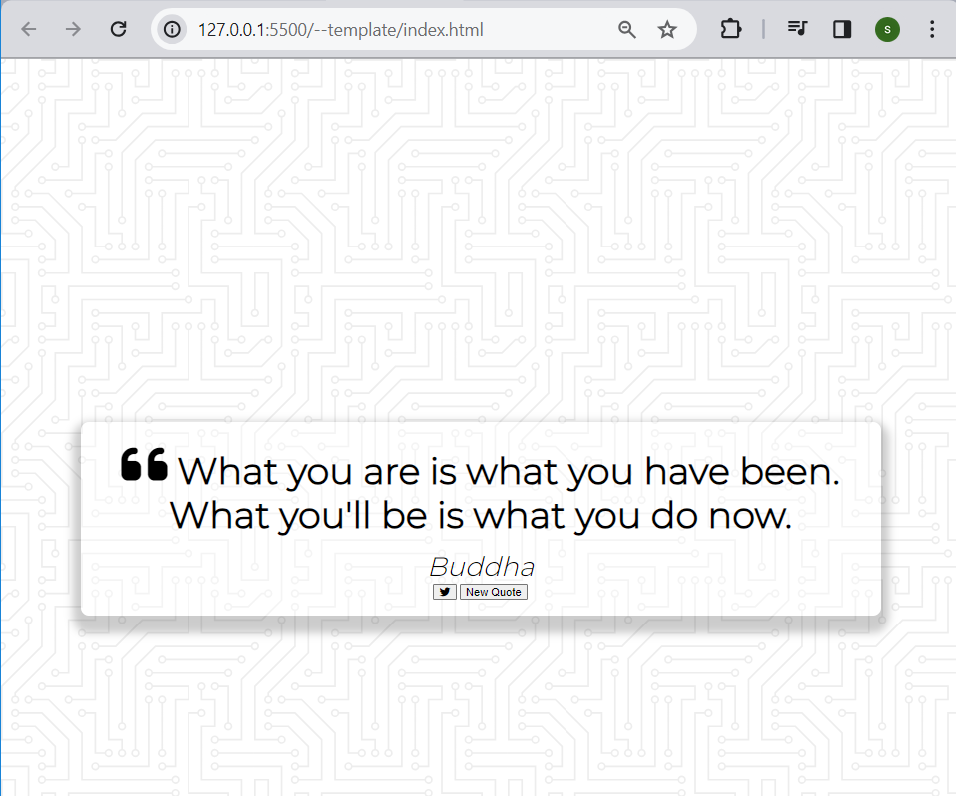
margin-top: 15px;

font-size: 2rem;

font-weight: 400;

font-style: italic;

}



6.When changed to a mobile device it doesn’t show any margin. So we need to add that.

/\* Media Query: Tablet or smaller \*/

@media screen and (max-width:1000px){

.quote-container{

margin:auto 10px;

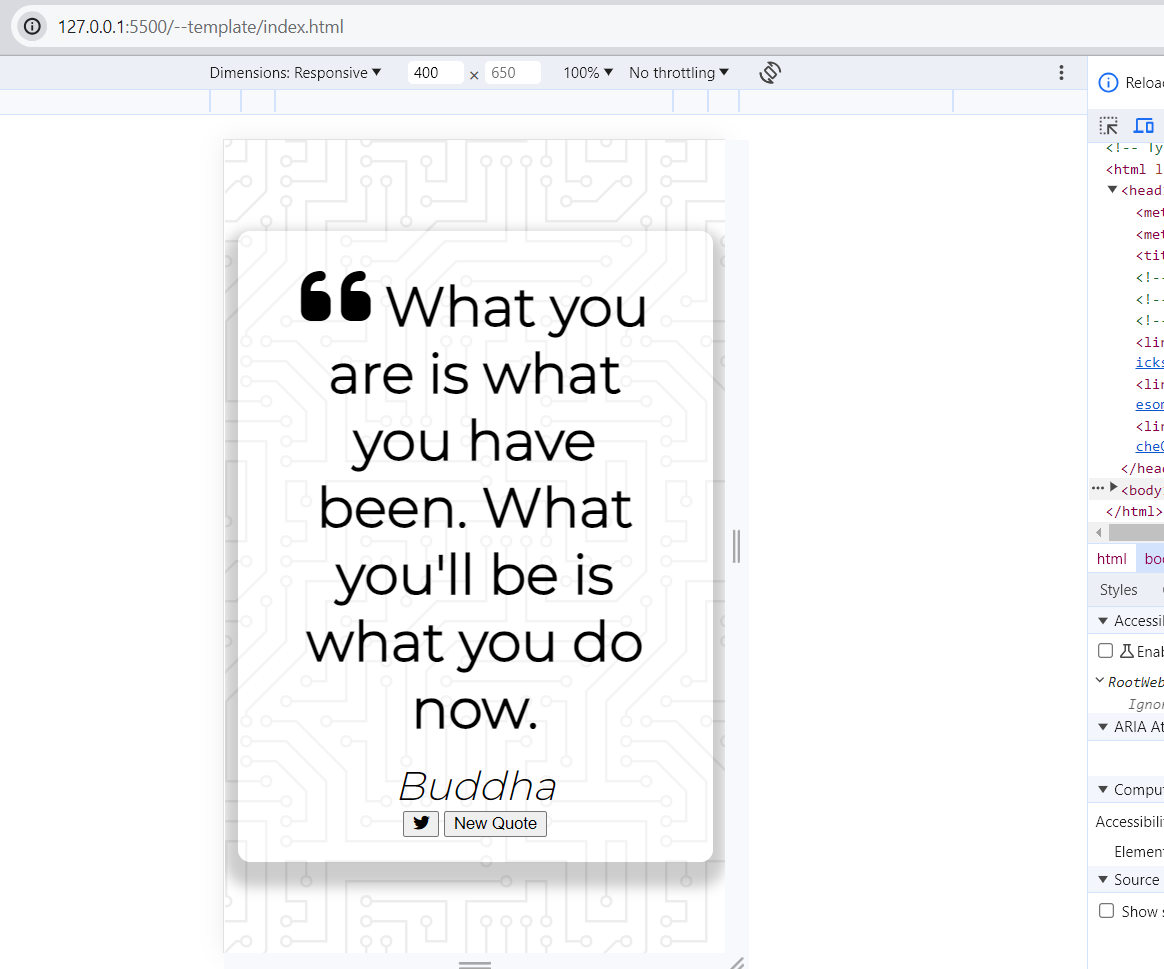
}

.quote-test{

font-size: 2.5rem;

}

}



7.Adding buttons to appropriate places by using flexbox:

Adding a space between buttons:

.button-container{

margin-top: 15px;

display: flex;

justify-content: space-between;

}

Stylising the button . And when user hovers he the pointer should change:

button {

cursor: pointer;

font-size: 1.2rem;

height: 2.5rem;

border: none;

border-radius: 10px;

color: #fff;

background: #333;

outline: none;

padding: 0.5rem 1.8rem;

box-shadow: 0 0.3rem rgba(121, 121, 121, 0.65);

}

When hovered over the button it should become brighter:

button:hover {

filter: brightness(110%);

}

Whenever the button is pressed it should move vertically down for sometime, then come back.

button:active {

transform: translate(0, 0.3rem); /\*in the vertical axis we are moving the button\*/

box-shadow: 0 0.1rem rgba(255, 255, 255, 0.65));

}

Note; hover and active are pseudo-class.

Make the twitter Icon little big and when hovered it should turn blue.

.twitter-button:hover {

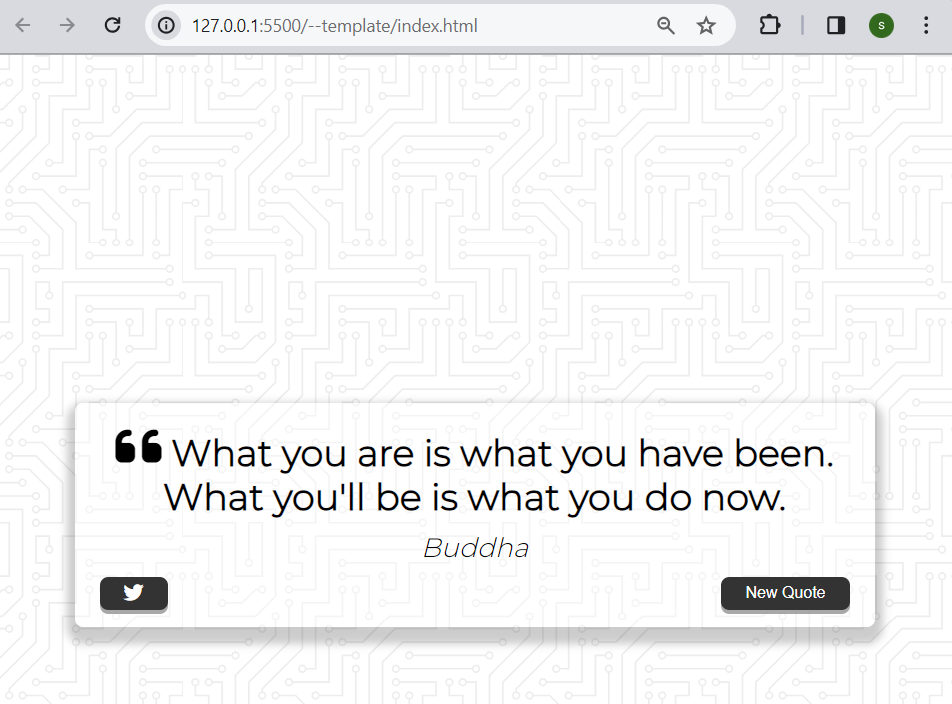
color: #38a1f3;

}

.fa-twitter {

font-size: 1.5rem;

}



8.In the JS file we need to add code for fetching the api value. Is a JSON file , which needs to be converted to json object. Then we can use try-catch here.

let apiQuotes = []; // global variable

//Get Quotes from API

async function getQuotes(){

const apiUrl= 'https://jacintodesign.github.io/quotes-api/data/quotes.json';

try {

const response = await fetch(apiUrl);

apiQuotes = await response.json();

console.log(apiQuotes[12]);

} catch (error){

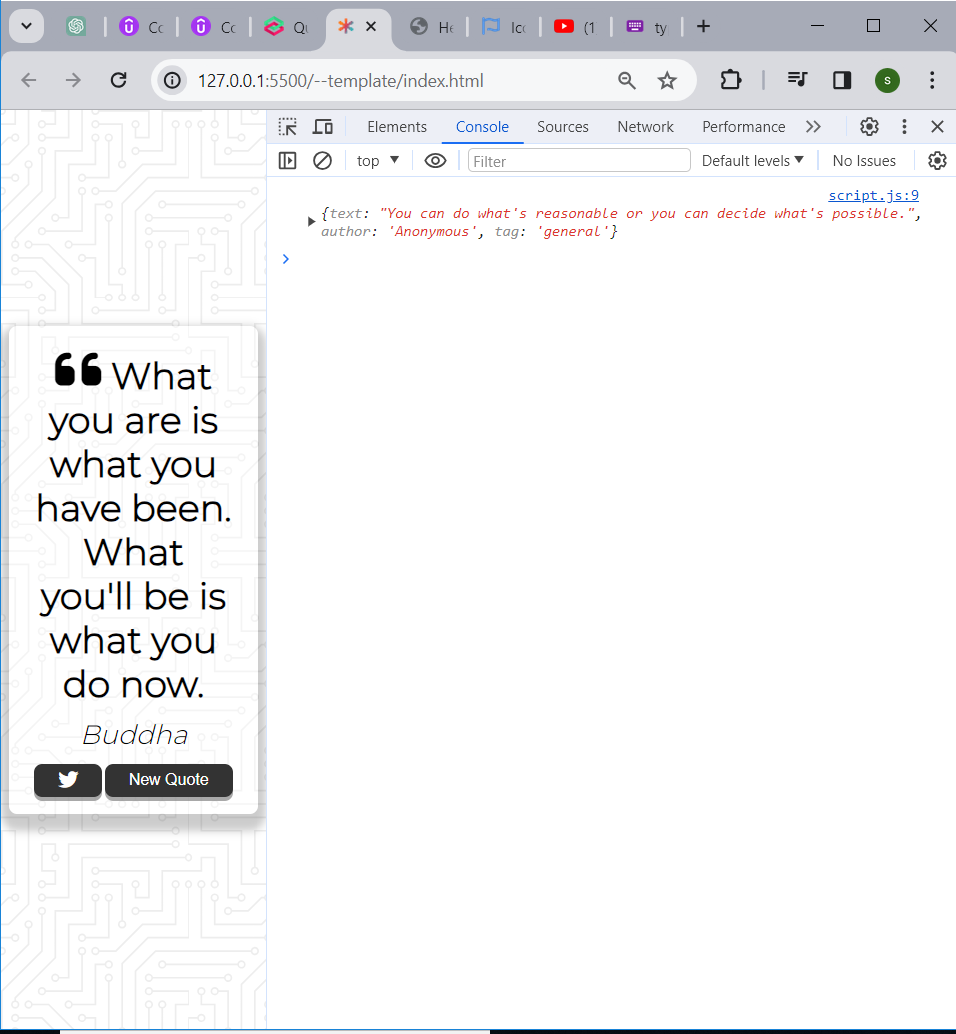
//Catch Error here

}

}

//On Load

getQuotes();



Here the 12th quote is picked. But we need to randomize this.

Call a function newQuotes inside try block: (This will randomly display values)  
//Show New Quote

function newQuote() {

//To pick a random quote from apiQuotes array

const quote = apiQuotes[Math.floor(Math.random() \* apiQuotes.length)];

console.log(quote);

}

Math.random gives values between 0 to 1. And apiQuotes.length gives the total number of quotes.

**Note:-** Steps to get the quotes from local quotes.js file

Step1: Write a quotes.js file with a const array of quotes, authors

const localQuotes = [

{

"text": "Genius is one percent inspiration and ninety-nine percent perspiration.",

"author": "Thomas Edison, type.fit"

},

{

"text": "You can observe a lot just by watching.",

"author": "Yogi Berra, type.fit"

},...

Step2:Link this with the html file

<script src="quotes.js"></script>

<script src="script.js"></script>

</body>

</html>

Step3:In the script.js use the console.log to display this localQuotes array(The below content is only required in script.sh file)

//Show New Quotes

function newQuote(){

//pick a random quote from localQuote array

const quote= localQuotes[Math.floor(Math.random()\*localQuotes.length)];

console.log(quote);

}

newQuote();

9.Updating the text and button fields in the project

Add constants for text and button fields in JS file  
// Creating constants

const quoteContainer = document.getElementById('quote-container');

const quoteText = document.getElementById('quote');

const authorTest = document.getElementById('author');

const twitterBtn = document.getElementById('twitter');

const newQuoteBtn = document.getElementById('new-quote');

Remove the text content already present in the html file

<!-- quote -->

<div class="quote-text">

<i class="fas fa-quote-left"></i>

<span id="quote"></span>

</div>

<!-- Author -->

<div class="quote-author">

<span id="author"></span>

</div>

Now modify the logging function to update author and phrase:

//Show New Quote

function newQuote() {

//To pick a random quote from apiQuotes array

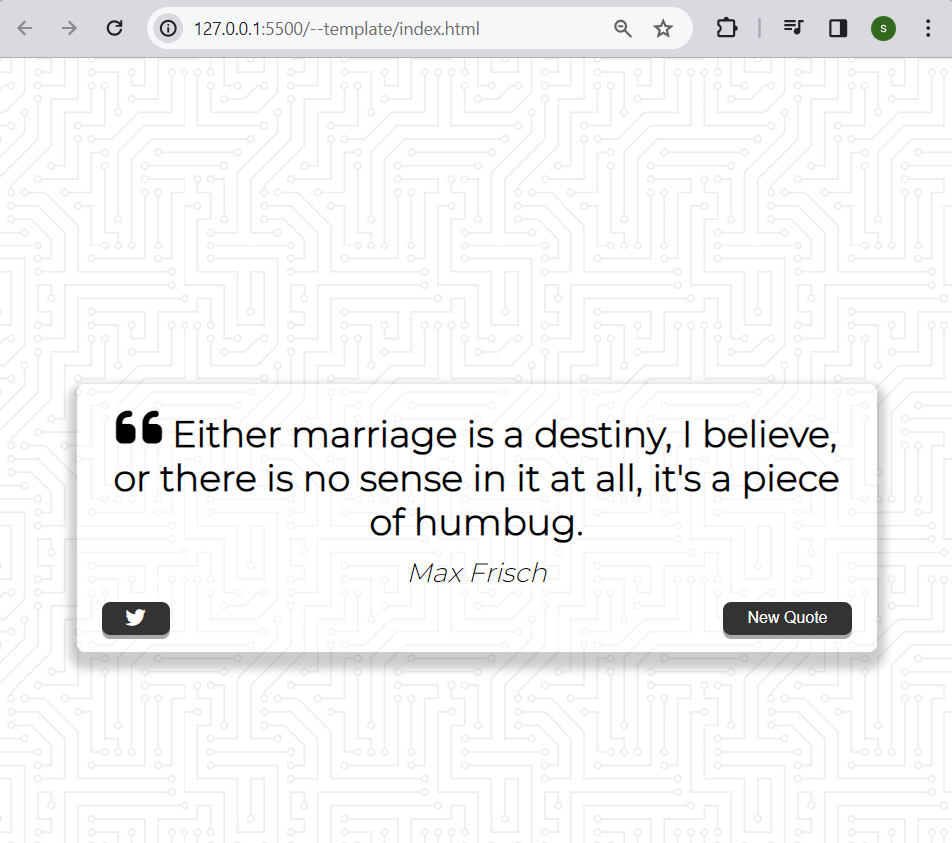
const quote = apiQuotes[Math.floor(Math.random() \* apiQuotes.length)];

//console.log(quote);

authorText.textContent = quote.author;

quoteText.textContent = quote.text;

}



If I refresh I can see the new quote and author.

10.If the author field is empty then we will fill it with “Unknown” and if the length of the Quote is large then we will reduce the font size

//Show New Quote

function newQuote() {

//To pick a random quote from apiQuotes array

const quote = apiQuotes[Math.floor(Math.random() \* apiQuotes.length)];

//console.log(quote);

//check if author field is empty. And replace with 'unknown'

if(!quote.author){

authorText.textContent = 'Unknown';

} else {

authorText.textContent = quote.author;

}

//check quote length to determine styling

if(quote.text.length > 50){

quoteText.classList.add('long-quote');

}else {

quoteText.classList.remove('long-quote');

}

quoteText.textContent = quote.text;

}

11.Creating twitter url with query. And then opening a new window for the created link  
link:https://twitter.com/intent/tweet

//Tweet Quote

function tweetQuote() {

const twitterUrl = `https://twitter.com/intent/tweet?text=${quoteText.textContent} - ${authorText.textContent}`; //back-ticks is used ==> template string. This helps to pass variable

window.open(twitterUrl, '\_blank');//\_blank ensures new page

}

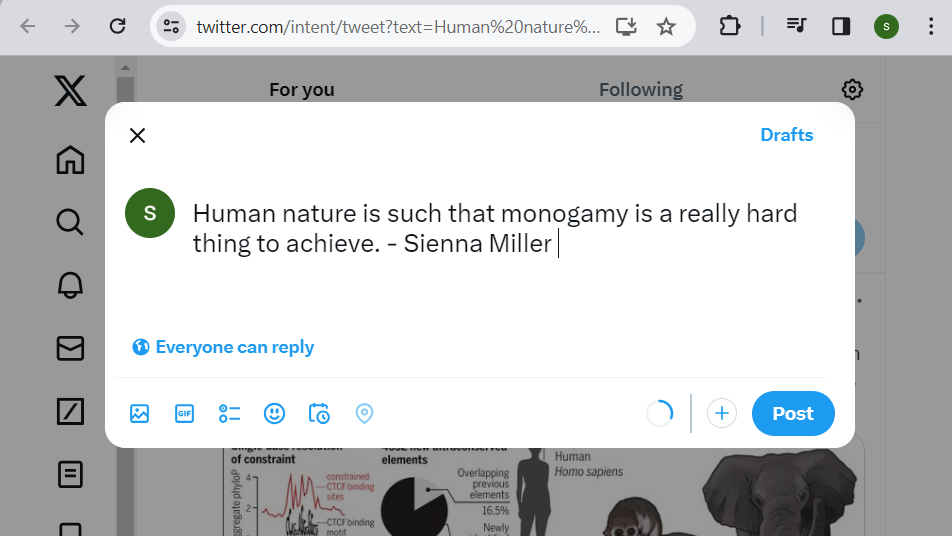
Adding in event listener:

//Event Listner

newQuoteBtn.addEventListener('click',newQuote);

twitterBtn.addEventListener('click',tweetQuote);

The twitter button is working:



Also the new-quote button is working fine.

12.Adding Loader

a)In HTML create a div:  
 <!-- Loader -->

<div class="loader" id="loader"></div>

<!-- link with JS -->

<!-- <script src="quotes.js"></script> -->

<script src="script.js"></script>

</body>

</html>

b)In CSS file add the content for animation of buffer

link:<https://www.w3schools.com/howto/howto_css_loader.asp>

/\* Loader \*/

.loader {

border: 16px solid #f3f3f3;

border-top: 16px solid #333;

border-radius: 50%;

width: 120px;

height: 120px;

animation: spin 2s linear infinite;

}

@keyframes spin {

0% { transform: rotate(0deg); }

100% { transform: rotate(360deg); }

}

c) In the java script create two functions. One for hiding the quote and other for hiding the spinner.

const loader = document.getElementById('loader');

let apiQuotes = []; // global variable array

//show loading

function loading() {

loader.hidden = false;

quoteContainer.hidden = true;

}

//Hide loading

function complete() {

quoteContainer.hidden = false;

loader.hidden = true;

}

Then call the loading() function at the start of

async function getQuotes(){

loading();

function newQuote() {

loading();

And the complete() function at the end of :

function newQuote() {

loading();

……

complete();

}

This creates a spinner when there is buffering in the page.

Final screen:

