

Low Level Design (LLD)

Low Level Design (LLD)

Random JokesApp

ASIBUL HASAN

|  |  |
| --- | --- |
| Table of contents | |
| 1. Prerequisites | 3 |
| 2. Objectives | 3 |
| 3. Random joke generator steps | 3 |
| 3.1 Basic structure of the joke generator | 4 |
| 3.2 Adding a title | 5 |
| 3.3 Set up the joke viewing zone | 6 |
| 3.4 Set up the generate joke button | 7 |
| 3.5 Activating the button | 8 |
| 4. Final Output |  |
| 5. Conclusion |  |



Low Level Design (LLD)

1.Prerequisites

To follow this tutorial, the reader should have the following:

* Essential expertise in HTML, CSS, and JavaScript.
* A code editor, preferably [Visual Code Studio](https://code.visualstudio.com/).

2.Objectives

By the end of this tutorial, learners should be able to:

* Design a basic HTML container to contain content.
* Connect API links to get data from other web pages.
* Assign actions to HTML buttons with eventListeners in javaScript.
* Develop a complete functional random joke generator.

3.Random joke generator steps

This project will generate random jokes using API links.

First, we will make a box on the web page and add a title. Then we will create an area in which all the joke generators will be seen.

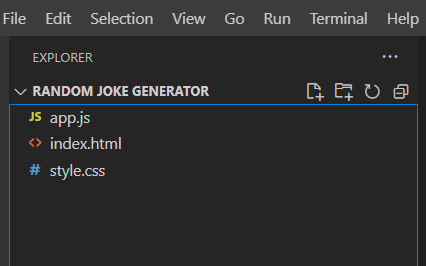
Below the area will be a generate button that will generate a different joke each time it is clicked on.

We will not use any text manually. Here we will use an API link to collect all the information from other places with the help of the fetch method and then display it on the webpage with the help of textContent.

Create the following files:

* index.html
* style.css
* app.js

This is the file structure.



**3.1: Basic structure of the joke generator**

We will start by setting up the basic structure of the joke generator. The background-color of our basic box and it will be 550px wide. We will use a border radius of 5px to round the edges of the box. Refer to the snippet below:

<!DOCTYPE html>

<html>

  <head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" type="text/css" href="./style.css" />

    <title>Random Joke Generator Using javascript</title>

</head>

  <body>

      <!-- container -->

      <div class="container">

        <!-- heading -->

        <h1>Jokes</h1>

        <!-- joke text -->

        <p class="joke-text">

          Joke Text Goes In Here...

        </p>

        <!-- buttons -->

        <div class="buttons">

          <!-- .new-joke Button -->

          <button class="btn new-joke-btn">New Joke</button>

        </div>

      </div>

    <script type="text/javascript" src="./app.js"></script>

  </body>

</html>

The box will be on the top center position of our page by setting its position

\* {

\* {

    padding: 0;

    margin: 0;

    box-sizing: border-box;

  }

  a {

    color: #111;

    text-decoration: none;

}

3.2: Adding a title

The next step is to add a title.

The title is there to enhance the page’s appearance

/\* body \*/

  body {

    width: 100vw;

    height: 100vh;

    overflow: hidden;

    background-image: radial-gradient(circle, #d16ba5, #c777b9, #ba83ca, #aa8fd8, #9a9ae1, #8aa7ec, #79b3f4, #69bff8, #52cffe, #41dfff, #46eefa, #5ffbf1);

    font-family: sans-serif;

    display: flex;

    justify-content: center; /\* horizontally center \*/

    align-items: center; /\* vertically center \*/

    text-align: center;

  }

  /\* container \*/

  .container {

    width: 600px;

    height: fit-content;

    padding: 50px 20px; /\* top-bottom left-right \*/

    background-image: linear-gradient(to right bottom, #051937, #004d7a, #008793, #00bf72, #a8eb12);

    border-radius: 29px;

}

**3.3 Set up the joke viewing zone**

Moving on to the third task, we will create an area to load all our jokes. This will be made possible by using <p> tags.

There is no need for a specific height and its size will be automatically determined based on the size of the content in it. It will have opacity set to 0 to visually hide everything in this section.

<div id="joke-section"></div>

h1 {

    font-size: 10vw; /\* make our h1 tag larger \*/

    font-family: sans-serif; /\* choosing our font \*/

    background: linear-gradient(to right, rgba(255, 215, 255, 0) 0%, rgba(225, 255, 255, 0.5) 20%, rgba(255, 255, 255, 0) 61%), linear-gradient(rgb(97, 183, 217) 52%, rgb(224, 246, 255) 60%, rgb(78, 99, 132) 61%); /\* you can change the colors based on your preference \*/

    background-clip: text; /\*it defines how far the background should extend within an element, here we set it to text \*/

    -webkit-background-clip: text; /\*for browsers compatibility \*/

    -webkit-text-fill-color: transparent; /\* specifies the fill color of text characters. We use transparent to use the background as our text fill  \*/

    animation: wave 2000ms ease alternate infinite;

    transition: all 0.4s ease;

    }

Output:



**3.4: Set up the generate joke button**

This button will help us generate random jokes when clicked. We will design the button using the code snippet below.

<input type="submit" id="btn" value="Generate Jokes" />

}

  .btn {

    font-family: Teko;

    font-weight: 0;

    font-size: 16px;

    color: #ffffff;

    background: linear-gradient(90deg, #0066CC 0%, #d62828 100%);

    padding: 10px 47px;

    border: none;

    box-shadow: rgb(0, 0, 0) 0px 0px 0px 0px;

    border-radius: 22px 50px;

    transition : 327ms;

    transform: translateY(0);

    }

  .btn:hover{

    transition : 327ms;

    padding: 10px 50px;

    transform : translateY(-0px);

    background: linear-gradient(90deg, #0066CC 0%, #d62828 100%);

    color: #ffffff;

    border: none;

    }

**3.5 Activating the button**

Since we have the required layout, it’s time to implement this example with the help of javaScript. First, we must get the joke container and then generate joke button, as shown below.

// grab a reference for necessary HTML elements

// .joke-text

const jokeText = document.querySelector('.joke-text');

// .new-joke-btn

const newJokeBtn = document.querySelector('.new-joke-btn');

// .tweet-btn (link)

const tweetBtn = document.querySelector('.tweet-btn');

// add 'click' eventListener to .new-joke-btn

newJokeBtn.addEventListener('click', getJoke);

The next thing we have to do is include an API in our project to fetch content from other websites. Refer to the following code snippet.

getJoke();

// getJoke() function definition

function getJoke() {

  // make an API request to https://icanhazdadjoke.com/'

  fetch('https://icanhazdadjoke.com/', {

    headers: {

      'Accept': 'application/json'

    }

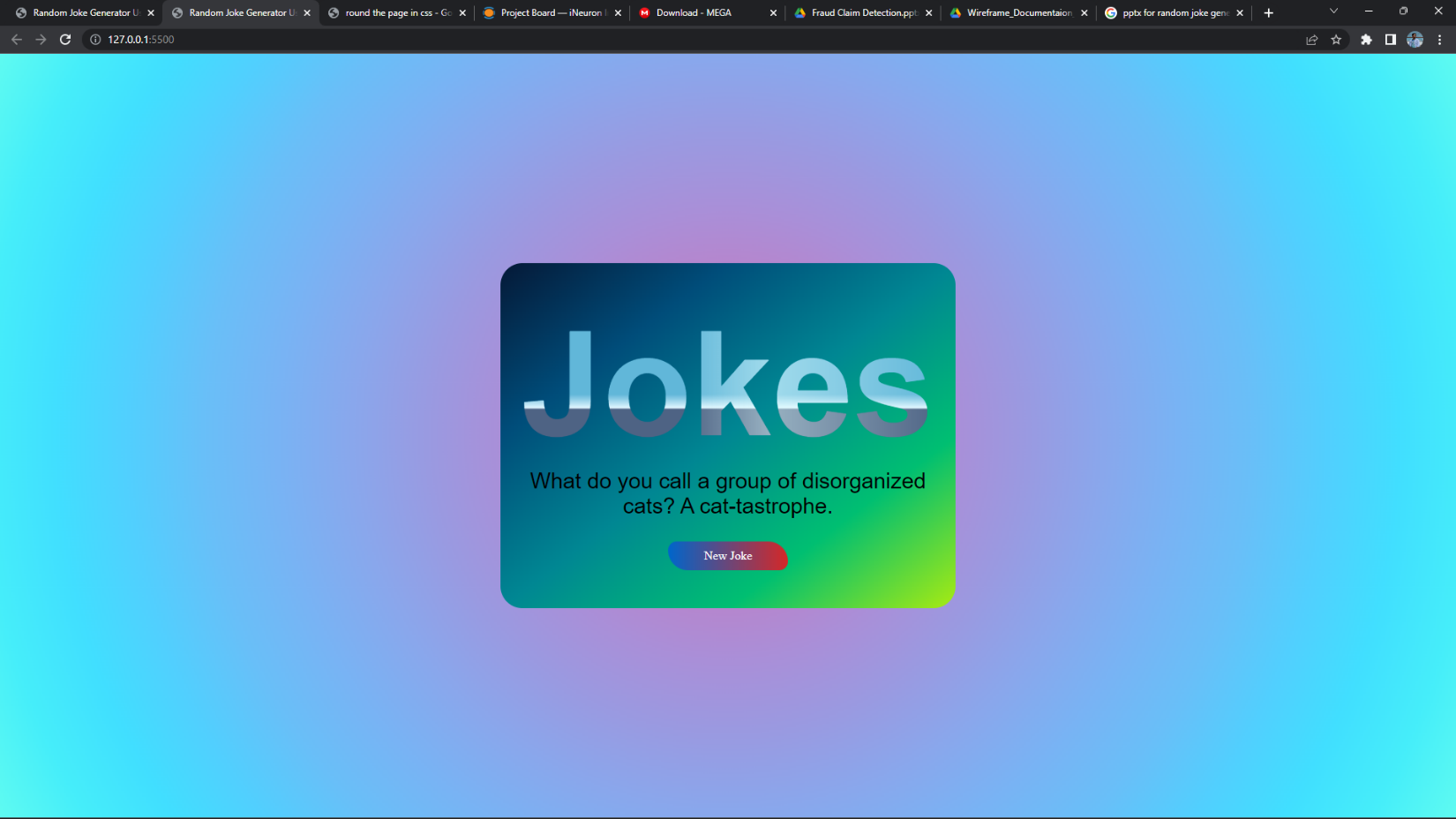
  }

Next, we will code a method and name it acquireJoke. This method will be ideal since it will help us achieve the following:

* First, we will eliminate class .content-fade, ensuring that nothing can be viewed with the help of classList.remove.
* The data obtained from the API using the fetch method will be arranged to display with the help of textContent.
* We will create a display using paragraph tags and add class .content-fade using classList. As a result, our text will be seen in the display.
* ).then(function(response) {
* /\* convert Stringified JSON response to Javascript Object \*/
* return response.json();
* }).then(function(data) {
* /\* replace innerText of .joke-text with data.joke \*/
* // extract the joke text
* const joke = data.joke;
* // do the replacement
* jokeText.innerText = joke;
* });

The last part is to assign the method we just created to display a joke whenever our button is clicked.

4**.Final Output:**



**5.Conclusion:**

We have covered using API links to bring material from another website to your website and develop a random joke generator in javaScript.