DOCUMENTATION OF IMPLEMENTATION

Creating a Google search homepage clone using HTML, CSS, and JavaScript is a great project to improve our web development skills. Here's a step-by-step of documentation for implementing this project:

Step 1: Project Setup

- Creating a new project folder and name it something like "Google Clone."
- Inside this folder, I made my project structure with the following files and folders:
    - **index.html**: The main HTML file for the homepage.
    - **style.css**: The CSS file for styling the page.
    - **script.js**: The JavaScript file for adding functionality.
    - **images/**: A folder to store any images you may need.

Now, to reduce the file size of the .js file, I inculcated the script.js part, i.e., the JavaScript part inside the index.html file only for this basic level project as it wasn’t requiring much use of JavaScript in this.

Step 2: HTML Structure

- I began by creating the basic HTML structure for my homepage. I used semantic HTML elements to structure the page properly.
- In this, I created various class to assign different elements at different places. Along with that also anchored many texts through anchor tag <a> which was again a crucial part in HTML designing.
- The logo or Google main image is also put in this index file for it’s appearance on the webpage.

Step 3: CSS Styling and Responsiveness

- Then I styled my Google clone using CSS to make it visually similar to the Google homepage. I made the use of my own creativity for this, but also made sure to include responsiveness for different screen sizes.
- Also, there I needed to use CSS Flexbox for layout, and CSS transitions/animations for interactive elements like buttons and links.

Step 4: JavaScript Functionality

- As mentioned earlier, for this project I inculcated the JavaScript inside the index.html file as there wasn't much of need to create a separate file for the same. The use of <script> tag was done for the same.
- I implemented JavaScript functionality to handle user interactions, such as searching and navigation.
- Then created a JavaScript function to handle the search functionality. This function listens for user input, execute the search, and display the search results.
- Also, implemented the additional functionality, such as toggling the navbar, displaying suggestions as the user types, etc. Plus, all the buttons on the page are fully connected to their respective links same as in real webpage.

Step 5: Modulation of code

- Ensured that my code is modular by breaking it down into smaller functions and files for better maintainability.
- I added comments in code thoroughly to explain the purpose of different functions, variables, and sections. This made it easier for you and others to understand and modify the code in the future.

Step 6: Live Site Deployment

- My webpage is then deployed to a web hosting service such as GitHub Pages.
- I made sure to update the links to my CSS and tags of JavaScript files in the HTML to reflect the live URLs.
- Then testing was done for the deployed site to ensure everything is working as expected.
- I created a GitHub repository for my project named as “Liveasy-project”
- Then uploaded my project files to the repository, and made sure to push updates as my work on the project.

Step 7: Final Testing

- Tested my Google clone thoroughly on various devices and browsers to ensure cross-browser compatibility and responsiveness.
- I encountered and corrected the bugs or issues faced during testing.

Aryan Shrivastava