SECURITY RISK ASSESSMENT REPORT

Code Review – Home Page

https://sit-chameleon-website-0bc2323.ts.r.appspot.com/

VERSION 0.0.1

20/11/2023

Miriam Azmy

TABLE OF CONTENTS

1. Code to be Reviewed:

```
import chameleonLogo from "./assets/Header-Chameleon.png";
import cityOfMelProject from "./assets/Thumbnail-CoM.png";
import evProject from "./assets/Thumbnail-EV.jpeg";
import websiteProject from "./assets/Thumbnail-Website.png";

const content1 =
 "At Chameleon, our mission is to research, create, test, document and deploy IoT-based solutions to enhance life through the application of smart city technologies including: the building of smarter cities, homes, transportation, and energy management systems.";
const content2 =
 "Lorem, ipsum dolor sit amet consectetur adipisicing elit. Soluta possimus numquam atque odio ab suscipit ipsam reiciendis alias, facere enim.";
const content3 = "City of Melbourne Open Data";
const content4 =
 "Lorem, ipsum dolor sit amet consectetur adipisicing elit. Soluta possimus numquam atque odio ab suscipit ipsam reiciendis alias, facere enim.";
const content5 = "Website Uplift";
const content6 =
 "Lorem, ipsum dolor sit amet consectetur adipisicing elit. Soluta possimus numquam atque odio ab suscipit ipsam reiciendis alias, facere enim.";

const Homepage = () => {
 return (
  <>
   <header className="bg-pewter p-2">
    <div className="container flex flex-col align-center md:flex-row md:justify-between p-4 gap-4">
     <img
      src={chameleonLogo}
      alt=""
      style={{ width: "269px", height: "269px" }}
      className="mx-auto"
     />
     <div className="my-auto">
      <p className="align-middle flex-1 text-xl">{content1}</p>
     </div>
    </div>
   </header>
   <section className="bg-green-emrld">
    <div className="">
     <h2 className="text-center pt-3 font-bold">Our Projects</h2>
     <div className="flex flex-col items-center justify-center lg:flex-row text-center lg:justify-around gap-4 p-5 container">
      <div class="max-w-sm rounded overflow-hidden shadow-lg bg-green-sage">
       <img
        class="w-full h-40 max-w-full object-cover"
        src={evProject}
```

```
            alt="Electric vehicle charging station"
          />
          <div class="px-6 py-4">
            <div class="font-bold text-xl mb-2">EV Adoption</div>
            <p class="text-gray-700 text-base">{content2}</p>
          </div>
        </div>
        <div class="max-w-sm rounded overflow-hidden shadow-lg bg-green-sage">
          <img
            class="w-full h-40 max-w-full object-cover"
            src={cityOfMelProject}
            alt="Melbourne street"
          />
          <div class="px-6 py-4">
            <div class="font-bold text-xl mb-2">{content3}</div>
            <p class="text-gray-700 text-base">{content4}</p>
          </div>
        </div>
        <div class="max-w-sm rounded overflow-hidden shadow-lg bg-green-sage">
          <img
            class="w-full h-40 max-w-full object-cover"
            src={websiteProject}
            alt="Chameleon"
          />
          <div class="px-6 py-4">
            <div class="font-bold text-xl mb-2">{content5}</div>
            <p class="text-gray-700 text-base">{content6}</p>
          </div>
        </div>
      </div>
    </section>
  </>
 );
};

export const homeSearchableContents = [
 content1,
 content2,
 content3,
 content4,
 content5,
 content6,
];

export default Homepage;
```

2. Purpose of the Code

The provided code snippet serves the purpose of constructing a responsive navigation bar within a web application. At its core, the code aims to enhance the user interface by incorporating essential elements such as a logo, a toggle button, and navigation links.

The `<img>` element is responsible for displaying the logo, identified by the variable `chameleonHeader`. The `alt` attribute, although currently empty, indicates that the image is intended for non-visual contexts, promoting accessibility. The image is configured to have a fixed height and width of 70 pixels, adhering to a specific design standard.

The toggle button, represented by the `<button>` element, employs the popular `FaBars` icon from a library, which implies a hamburger menu design commonly used for mobile responsiveness. The button is styled using Tailwind CSS classes to ensure consistent styling across different screen sizes. The `onClick` event handler toggles the `navbarOpen` state, suggesting that it plays a role in controlling the visibility of the navigation menu for smaller screens.

The navigation menu itself is structured within a `<div>` element with a conditional class, responding to the value of the `navbarOpen` state. This responsiveness is crucial for adapting the layout based on the user's device, offering an optimal experience.

Within the menu, links to "Home," "Projects," and "News" are implemented using `<a>` elements. These links are styled for visual appeal and user interaction, with uppercase text, bold fonts, and a subtle hover effect for improved user feedback. The navigation links follow best practices for web development, providing clear paths for users to navigate through different application sections.

In summary, the code snippet contributes to the creation of a visually appealing and functionally responsive navigation bar, fostering a positive user experience within the broader context of a web application.

3. Code Structure and Organization

The code has a clean and simple structure, with major elements like the logo, toggle button, and navigation links contained within well-defined containers. The usage of conditional classes to govern responsiveness improves organisation and ensures an orderly presentation across several device sizes. This streamlined structure improves code readability and maintainability, creating a positive development experience and making future changes or additions easier.

4. Use of Data Structures

The code primarily focuses on presenting a user interface and doesn't involve the use of intricate data structures. Instead, it centers around JSX elements, such as images, buttons, and links, to create a visually appealing navigation bar. Styling is achieved using Tailwind CSS classes, emphasizing simplicity and ease of maintenance. This absence of complex data structures aligns with the code's purpose of providing a clear and concise representation of the frontend, promoting readability and facilitating straightforward design modifications.

5. Commenting and Documentation

Enhancing code maintainability is crucial, and in this context, incorporating comments and documentation would greatly contribute to the overall code quality. Providing clear comments elucidates the purpose of various elements and sections, facilitating comprehension for developers who may not be intimately acquainted with the codebase. By adopting these practices, we fortify collaboration, reduce onboarding complexities for new team members, and establish a foundation for a resilient and scalable codebase.

6. Coding Standards Adherence

The code adheres to the Tailwind CSS styling conventions. Ensure consistency in class names and styling throughout the codebase.

7. Code Complexity

The code is relatively simple, with a focus on styling and layout. Consider simplifying the structure of the nested <ul> elements to avoid redundancy.

8. Potential Problem Areas

No urgent or critical problems that effects the overall quality of the page or functionality of the website.

9. Security and Performance Checks

From a security perspective the provided code for the header appears to be focused on the frontend presentation and user interface, it's not immediately clear if the code adheres to specific security best practices. To enhance security the following should be considered:

Image Source (chameleonHeader):

Ensure that the image source (chameleonHeader) is validated and doesn't allow for potential security risks like injection attacks. If this source is dynamically generated or fetched from user inputs, proper validation is crucial to prevent issues like XSS (Cross-Site Scripting) attacks.

Button Click Handler (onClick={() => setNavbarOpen(!navbarOpen)}):

The button click handler appears to be related to toggling the navbarOpen state. This is a common practice for responsive navigation. Ensure that the state changes are handled securely and that there are no vulnerabilities associated with state manipulation.

Communication with Backend:

If this is just a frontend component, consider how it interacts with the backend. Ensure that data exchanged between the front end and back end is handled securely, and that there are no opportunities for injection attacks.

Data Validation:

If the code deals with user inputs (e.g., URLs, parameters), ensure that proper data validation and sanitization are in place to prevent security issues.

To comprehensively assess the security of the entire application, it's important to review the backend code, server configurations, and overall architecture. This code snippet alone doesn't provide a complete picture of the application's security posture.

10. Proposed changes to Code

```
<img
 src={chameleonHeader}
 alt=""
 style={{ height: "70px", width: "70px" }}
/>

<button
 className="cursor-pointer text-xl leading-none px-3 py-1 border border-solid border-transparent rounded bg-transparent block lg:hidden outline-none focus:outline-none"
 type="button"
 onClick={() => setNavbarOpen(!navbarOpen)}
>
 <FaBars />
</button>

</div>

<div
 className={
  "lg:flex flex-grow items-center transition duration-500 ease-in-out" +
  (navbarOpen ? " flex" : " hidden")
 }
 id="example-navbar-danger"
>

<ul className="flex flex-col mx-auto lg:flex-row list-none lg:ml-auto ">
 <li className="nav-item">
  <a
   className="px-3 py-2 flex items-center uppercase font-bold leading-snug hover:opacity-75 no-underline"
   href="/"
  >
   Home
  </a>
 </li>
 <li className="nav-item">
  <a
   className="px-3 py-2 flex items-center uppercase font-bold leading-snug hover:opacity-75 no-underline"
   href="/projects"
  >
   Projects
  </a>
 </li>

 {/* Remove duplicate nested ul */}
 <li className="nav-item">
  <a
```

```
    className="px-3 py-2 flex items-center uppercase font-bold leading-snug hover:opacity-75 no-underline"
    href="/news"
   >
    News
   </a>
  </li>
</ul>
</div>
```

11. Findings

Changes made:

Removed Duplicate Nested <ul>:
Removed the duplicate nested <ul> element to avoid redundancy and maintain a cleaner structure.

Improved Code Comments (Not Explicitly Shown in the Code):
Consider adding comments to clarify the purpose of certain elements or sections, especially for developers who may not be familiar with the code. For example, you could add comments to describe the purpose of the <img> tag, the toggle button, and the navigation links.