SECURITY RISK ASSESSMENT REPORT

Code Review – Login

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

VERSION 0.0.1

23/11/2023

Miriam Azmy

TABLE OF CONTENTS

1. Code to be Reviewed

```
import React, { Component } from "react";
import Background from "./image/Background.png";
import Logo from "./image/Chameleon_Logo.png";
import Google from "./image/google.png";
import Linkedin from "./image/linkedin.png";
import Microsoft from "./image/microsoft.png";

class Login extends Component {
  render() {
    return (
      <div
        style={{
          height: "100vh",
          backgroundImage: `linear-gradient(to right, transparent 43.43%, white 0%), url(${Background})`,
          backgroundSize: "cover",
          backgroundPosition: "left center",
          overflow: "hidden",
        }}
      >
        <div
          className="container mx-auto p-0"
          style={{ maxWidth: "80%", marginTop: "3%", border: "3px solid gray" }}
        >
          <div className="grid grid-cols-12">
            <div
              className="md:col-span-5"
              style={{
                backgroundColor: "gray",
                display: "flex",
                flexDirection: "column",
                justifyContent: "center",
                alignItems: "center",
              }}
            >
              <img
                src={Logo}
                alt="Logo"
                style={{ width: "50%", height: "auto", marginBottom: "2rem" }}
              />
              <p
                style={{
                  fontSize: "1.7rem",
                  fontWeight: "bold",
                  textAlign: "center",
                }}
              >
                Enhancing life through IoT-Powered Smart City Solutions
              </p>
              <div
                style={{
                  position: "absolute",
                  right: 0,
                  top: 0,
                  bottom: 0,
                  width: "1px",
                  backgroundColor: "black",
                }}
              ></div>
            </div>

            <div className="md:col-span-7" style={{ padding: "8%" }}>
              <h1
                style={{
                  fontWeight: "bold",
                  marginBottom: "2rem",
                  textAlign: "center",
                }}
              >
                Login
              </h1>

              <form>
                <div controlId="formEmail" style={{ marginBottom: "2rem" }}>
                  <input
                    type="Email"
                    placeholder="EMAIL ADDRESS"
                    className="w-100"
                    style={{
                      backgroundColor: "#ccc",
                      border: "1px solid black",
                      height: "50px",
                    }}
                  />
                </div>

                <div controlId="formpassword" style={{ marginBottom: "2rem" }}>
                  <input
                    type="password"
                    placeholder="PASSWORD"
                    className="w-100"
                    style={{
                      backgroundColor: "#ccc",
                      border: "1px solid black",
                      height: "50px",
                }}
              />
            </div>

            <p style={{ fontWeight: "bold", textAlign: "right" }}>
              <a href="/signup">Sign-up?</a>
              <br />
              <a href="/reset">Forgot password?</a>
            </p>

            <div className="d-flex justify-content-center mb-3">
              <button
                className="bg-green-emerald p-3 text-white w-24"
                type="submit"
                style={{ padding: "10px 80px", fontSize: "1rem" }}
              >
                {" "}
                LOGIN
              </button>
            </div>

            <div className="d-flex justify-content-between">
              <div
                style={{
                  backgroundColor: "green",
                  borderRadius: "50%",
                  width: "40px",
                  height: "40px",
                  display: "flex",
                  justifyContent: "center",
                  alignItems: "center",
                }}
              >
                <img src={Linkedin} alt="LinkedIn Icon" />
              </div>

              <div
                style={{
                  backgroundColor: "green",
                  borderRadius: "50%",
                  width: "40px",
                  height: "40px",
                  display: "flex",
                  justifyContent: "center",
                  alignItems: "center",
                }}
              >
                <img
                  src={Google}
```

```
                      alt="Google Icon"
                      style={{ width: "60%", height: "60%" }}
                    />
                  </div>

                  <div
                    style={{
                      backgroundColor: "green",
                      borderRadius: "50%",
                      width: "40px",
                      height: "40px",
                      display: "flex",
                      justifyContent: "center",
                      alignItems: "center",
                    }}
                  >
                    <img
                      src={Microsoft}
                      alt="Microsoft Icon"
                      style={{ width: "60%", height: "60%" }}
                    />
                  </div>
                </div>
              </form>
            </div>
          </div>
        </div>
      </div>
    );
  }
}

export default Login;


// import React, { Component } from 'react';
// import 'bootstrap/dist/css/bootstrap.min.css';
// import Background from './image/Background.png';
// import Logo from './image/Chameleon_Logo.png';
// import Google from './image/google.png';
// import Linkedin from './image/linkedin.png';
// import Microsoft from './image/microsoft.png';
// import { Container, Row, Col, Form, Button } from 'react-bootstrap';
// import { signInWithGooglePopup, createUserDocFromAuth, signinAuthUserWithEmailAndPassword } from
'./utils/firebase'


// class Login extends Component {
//   render() {
```

```
//    const Login = (props) => {

//      const nanvigate = useNavigate();

//      const logGoogleUser = async () => {
//        const { user } = await signInWithGooglePopup();
//        const userDocRef = await createUserDocFromAuth(user)
//      }

//      const [contact, setContact] = useState({
//        email: '',
//        password: '',
//      })

//      const { email, password } = contact

//      const handleChange = (event) => {

//        const { name, value } = event.target
//        setContact((preValue) => {
//          return {
//            ...preValue,
//            [name]: value
//          }
//        })
//      }

//      const handleSubmit = async (event) => {
//        event.preventDefault();
//        try {
//          const { user } = await signinAuthUserWithEmailAndPassword(username, password);
//          await createUserDocFromAuth(user, { email });
//        }
//        catch (error) {
//          console.log('error', error.message)
//        }
//      }


//    }

//    return (
//      <div style={{
//        height: '100vh',
//        backgroundImage: `linear-gradient(to right, transparent 43.43%, white 0%), url(${Background})`,
//        backgroundSize: 'cover',
//        backgroundPosition: 'left center',
//        overflow: 'hidden'
//        }}>
//        <Container fluid style={{ maxWidth: '80%', marginTop: '3%', border: '3px solid gray'}}>
//          <Row>
//              <Col md={5} style={{ backgroundColor: 'gray', display: 'flex', flexDirection: 'column', justifyContent:
'center', alignItems: 'center' }}>
//             <img src={Logo} alt="Logo" style={{ width: '50%', height: 'auto', marginBottom: '2rem' }} />
//               <p style={{ fontSize: '1.7rem', fontWeight: 'bold', textAlign: 'center' }}>Enhancing life through IoT-
Powered Smart City Solutions</p>
//               <div style={{ position: 'absolute', right: 0, top: 0, bottom: 0, width: '1px', backgroundColor: 'black'
}}></div>
//           </Col>

//           <Col md={7} style={{ padding: '8%' }}>
//             <h1 style={{ fontWeight: 'bold', marginBottom: '2rem', textAlign: 'center' }}>Login</h1>

//             <Form>
//              <Form.Group controlId="formEmail" style={{ marginBottom: '2rem' }}>
//               <Form.Control type="Email" placeholder="EMAIL ADDRESS" style={{ backgroundColor: '#ccc', border:
'1px solid black', height: '50px' }} />
//              </Form.Group>

//              <Form.Group controlId="formpassword" style={{ marginBottom: '2rem' }}>
//               <Form.Control type="password" placeholder="PASSWORD" style={{ backgroundColor: '#ccc', border:
'1px solid black', height: '50px' }} />
//              </Form.Group>

//              <p style={{ color: 'green', fontWeight: 'bold', textAlign: 'right' }}>
//               <a href="/signup">Sign-up?</a><br />
//               <a href="/forgot">Forgot password?</a>
//              </p>

//              <div className="d-flex justify-content-center mb-3">
//               <Button variant="success" type="submit" style={{ padding: '10px 80px', fontSize: '1rem' }}>
//                LOGIN
//               </Button>
//              </div>

//              <div className="d-flex justify-content-between">
//                 <div style={{ backgroundColor: 'green', borderRadius: '50%', width: '40px', height: '40px', display:
'flex', justifyContent: 'center', alignItems: 'center' }}>
//                 <img src={Linkedin} alt="LinkedIn Icon"/>
//               </div>

//                 <div style={{ backgroundColor: 'green', borderRadius: '50%', width: '40px', height: '40px', display:
'flex', justifyContent: 'center', alignItems: 'center' }}>
//                 <img src={Google} alt="Google Icon" style={{ width: '60%', height: '60%' }} />
//               </div>
//                    <div style={{ backgroundColor: 'green', borderRadius: '50%', width: '40px', height: '40px', display: 
'flex', justifyContent: 'center', alignItems: 'center' }}>
//                      <img src={Microsoft} alt="Microsoft Icon" style={{ width: '60%', height: '60%' }} />
//                    </div>
//                  </div>
//                </Form>
//              </Col>
//            </Row>
//          </Container>
//        </div>
//      );
//    }
// }

// export default Login;
```

2. Purpose of the Code

The purpose of this code is to create a modular and reusable React component that renders the UI for a login page. It integrates various images to enhance the visual appeal, including a background image, a distinctive Chameleon logo, and icons associated with popular social media platforms. This component is crucial for providing users with a seamless and visually pleasing authentication experience within the larger web application. In the live environment the code is non-functional and needs to be to the wider Deakin domain as it uses Deakin users' login.

3. Code Structure and Organization

The code follows "React's" component-based architecture and has a well-organized structure. It uses inline styles for layout and responsiveness, keeping stylistic issues within the component. The usage of a grid layout and clear separation of separate components, such as the Chameleon logo, description paragraph, and login form, shows a deliberate approach to code organisation and maintainability.

4. Use of Data Structures

The utilisation of data structures here is relatively straightforward, primarily revolving around the manipulation and presentation of static data within a React component. The code predominantly employs simple data structures, namely strings and objects, to facilitate the specification of image paths and the inline styling of elements.

Firstly, strings are employed for specifying image paths. The variables such as Background, Logo, Google, Linkedin, and Microsoft store string values representing the paths to corresponding image assets. These paths are essential for the img elements' src attributes, ensuring that the correct images are rendered in the user interface.

Secondly, objects are utilized for inline styling of elements within the component. The inline styles are defined directly within the JSX, using JavaScript objects. This practice allows for concise and localized styling, contributing to the component's modularity and maintainability. The styles define attributes such as height, backgroundImage, display, flexDirection, and more, shaping the visual appearance of the rendered elements.

React components, which are essentially JavaScript classes or functions, maintain their state natively, employing more complex data structures as needed. The focus in this snippet remains on using strings and objects for simple tasks, in keeping with the philosophy of keeping components modular and focused on their specialised roles.

5. Commenting and Documentation

A notable deficiency in the code is the absence of comments and documentation. To enhance collaboration and future maintenance, it's crucial to document the purpose and functionality of various elements, especially for developers who may not be familiar with the codebase. Comments should be strategically placed to clarify the intent behind specific sections and enhance overall code readability.

6. Coding Standards Adherence

While the code generally adheres to coding standards, there is room for improvement in terms of consistent indentation. Ensuring a uniform coding style across the entire codebase is essential for collaboration and code maintainability. Consistent indentation contributes to readability, making it easier for developers to understand the code's structure.

7. Code Complexity

The complexity of the code is moderate. The implementation of a grid layout and styling are relatively straightforward, making the code accessible to developers with a basic understanding of React and web development. However, the absence of comments may introduce challenges for those trying to comprehend specific functionalities or make modifications to the code.

8. Potential Problem Areas

One potential problem area is the lack of error handling in the login form. Robust error handling mechanisms should be implemented to provide users with meaningful feedback in case of authentication failures or other issues. Additionally, potential layout inconsistencies across different screen sizes should be addressed to ensure a seamless user experience.

9. Security and Performance Checks

In terms of its intended function, the offered code sample, which focuses on rendering a login page in a React application, appears to be secure. However, security concerns extend beyond the frontend code to include backend authentication, authorisation systems, and secure server connection. Because the code snippet only covers the visual display of the login page, a thorough security evaluation requires an examination of the broader application architecture and backend implementation.

The code follows several best practices in terms of best practices, such as using React components for modularity, inline styles for element styling, and importing image assets. However, there are areas for improvement and best practices to consider:

Secure Backend Implementation: Ensure that the backend authentication and user input handling follow best practices for security. This includes proper validation, sanitization, and protection against common security vulnerabilities like SQL injection and cross-site scripting (XSS).

Secure Communication: If the login form interacts with a backend API, ensure that the communication is secured using HTTPS to encrypt data in transit.

Form Validation: Implement client-side form validation to provide immediate feedback to users and reduce unnecessary requests to the backend for invalid data.

Error Handling: Enhance error handling within the code, providing users with clear error messages in case of authentication failures or other issues.

CSS-in-JS Libraries: Consider using CSS-in-JS libraries like styled-components or Emotion for a more dynamic and maintainable styling approach within React components.

Responsive Design: Ensure that the login page is responsive and usable across various devices and screen sizes.

Authentication: This is critical to the login functionality in web applications, and ensures a safe and secure log in which the current code does not define.

10. Proposed changes to Code

Several improvements can be made to enhance the code quality, maintainability, and user experience. The most critical change is the introduction of comments and documentation throughout the code. Comments should explain the purpose and functionality of different elements, providing valuable insights for developers working on the codebase. Additionally, refactoring for consistent indentation is recommended to ensure a uniform and readable coding style. Implementing error handling in the login form and addressing potential layout inconsistencies will contribute to a more robust and user-friendly solution.

```
import React, { useState } from "react";
import PropTypes from "prop-types";
import styled from "styled-components";
import { authenticateUser } from "./api/auth"; // Import secure authentication function

const LoginPage = () => {
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [error, setError] = useState(null);

  const handleLogin = async (event) => {
    event.preventDefault();
    try {
      // Call secure authentication function
      const user = await authenticateUser(email, password);
      // Handle successful authentication
    } catch (error) {
      // Handle authentication error
      console.error("Authentication error:", error.message);
      setError("Invalid email or password. Please try again.");
    }
  };

  return (
    <LoginPageContainer>
      <Logo src={LogoImage} alt="Chameleon Logo" />
      <LoginContent>
        <h1>Login</h1>
        <LoginForm onSubmit={handleLogin}>
          <FormInput
            type="email"
            placeholder="Email Address"
            value={email}
            onChange={(e) => setEmail(e.target.value)}
            required
          />
          <FormInput
            type="password"
            placeholder="Password"
            value={password}
            onChange={(e) => setPassword(e.target.value)}
            required
          />
          <ErrorMessage>{error}</ErrorMessage>
          <ForgotLinks>
            <ForgotLink href="/signup">Sign-up?</ForgotLink>
            <ForgotLink href="/reset">Forgot password?</ForgotLink>
          </ForgotLinks>
          <LoginButton type="submit">LOGIN</LoginButton>
        </LoginForm>
        <SocialIcons>
          <SocialIcon src={LinkedinIcon} alt="LinkedIn Icon" />
          <SocialIcon src={GoogleIcon} alt="Google Icon" />
          <SocialIcon src={MicrosoftIcon} alt="Microsoft Icon" />
        </SocialIcons>
      </LoginContent>
    </LoginPageContainer>
  );
};

// ... Styled components for improved readability

LoginPage.propTypes = {
  // Add PropTypes as needed for props
};

export default LoginPage;
```

State management has been improved by using React hooks, resulting in a more streamlined and efficient component. The provision of a separate state variable for error handling improves user feedback in the event of authentication failures. PropTypes contain proactive validation, which promotes code reliability. Reusable and encapsulated components for related UI elements have been developed, promoting modularity and ease of maintenance. The styled-components library was used to optimise styling, allowing for a more dynamic and maintainable approach. Furthermore, the use of a secure authentication method ('authenticateUser') for robust user authentication has been assumed. These enhancements attempt to improve the login page component's general readability, maintainability, and security while adhering to best practices and modern React development principles.

11. Findings

While the code successfully achieves its goal of rendering a visually appealing login page, the absence of comments and documentation is a significant drawback. These elements are crucial for promoting code understanding, collaboration, and long-term maintainability. The proposed changes aim to address these issues and further improve the overall quality of the code.