**Software Design Document**

**Project: Lutron**

**Introduction-**

By implementing troubleshooting feature, we intend to empower users to efficiently troubleshoot. The primary goal is to enhance overall user satisfaction and engagement by offering a reliable and intuitive mechanism for Troubleshooting login-related issues.

**System Architecture-**

* The **troubleshooting link** will be a part of the user interface on **the login page**.
* Consistent with best practices, the system architecture prioritizes **scalability and adaptability** for evolving user requirements.
* **Interactions involve server-side** (ASP.NET Core MVC) components.

**Components-**

**Controller**

* **Troubleshooting Controller:** Action method for rendering the troubleshooting view.

**View:**

* **Login.cshtml :** Layout for login page with an **integrated troubleshooting Link** (light brown colour)and pop-up.
* **Troubleshooting.cshtml :** Layout for Troubleshooting page with the customer support button (Dark blue colour)

**JavaScript**:

* Client side script handling **integrated troubleshooting Link** click events and pop-up Troubleshooting displays.
* Responsible for enhancing user interactivity and responsiveness.

**User Interface Design**

* **Login Form:** Integrate a troubleshooting Link (light brown colour) Below the login button on login page.

A screenshot of a login form

Description automatically generated

**OR Forget Password**

**Pop-up Design:**

* Delivers a **well-designed pop-up** interface triggered by the troubleshooting link, displaying user-friendly messages and suggestions.
* A user-friendly option for users to **easily close the pop-up**.

**A screenshot of a computer error

Description automatically generated**

**Interaction Flow**

* User clicks the **troubleshooting link**.
* **Client-side script** triggers call to the server.
* The server-side **troubleshooting Controller** responds with **troubleshooting feature**.
* The pop-up is dynamically displayed with **user friendly messages/suggestions**.

**Data Design**

* List of different **troubleshooting methods** Along with the **Customer Support** button

**Security Measures:**

* Ensures **secure data transfer** between the client and server using HTTPs to protect sensitive information during troubleshooting.
* Implements **server-side validation like** IsValidEmail, Password Strengthand **error-handling mechanisms** for enhanced security.

**Implementation Details**

* Use Asp.net core MVC to **create troubleshooting Controller and views**.
* Leverages **client-side JavaScript to handle user** interactions, including button clicks and asynchronous requests.

**Testing strategy**

* Unit testing-Verify the functionality of the troubleshooting link and pop-up.
* Integration testing-Ensure proper integration with existing login page.
* User Acceptance testing- **Test with users** to ensure the troubleshooting link meet there needs.

**Performance Considerations**

* **Test** the performance of the troubleshooting link and customer support button on different devices and browsers.

**Maintenance and Support**

* Include updates for any update to troubleshooting link.

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Role** | **Signature** | **Date** |
| Varun Bhatia | Reviewer |  |  |
| Varun Bhatia | Lead |  |  |
| Prabhash Sharma | Developer | Ptabhash | 24-01-2024 |