React Password Generator

Sushil Yadav

Document Version Control

Date Issued	Version	Description	Author
19-03-2023	1.2	Added Workflow chart	Sushil
24-03-2023	1.3	Added Exception Scenarios Overall, Constraints	Sushil
18-03-2023	1.1	First Draft	Sushil
06-05-2023	1.4	Added KPIs	Sushil
19-03-2023	1.5	Added user I/O flowchart	Sushil
28-05-2023	1.6	Restructure and reformat LLD	Sushil

Contents

Abstract

Users can create strong, random passwords using the React Password Generator, a web application created with the React framework. It gives different choices to customize the generated passwords, such as length, character kinds, and the presence or omission of confusing characters and uses the React library to build an intuitive and interactive user interface. The React Password Generator's primary role is to generate random passwords based on the user's settings. The project also includes tools like a password strength meter that assesses the produced password and provides feedback on its robustness. Users that need to establish strong and secure passwords for their online accounts can benefit from the React Password Generator project, which offers a useful tool for protecting users' personal information. Overall, the project contributes to a safer online environment for people and businesses by showcasing the strength and adaptability of the React framework in creating flexible and feature-rich web applications.

1 Introduction

1.1 Why this Low-Level Design Document?

Components:

- App: The main component that serves as the entry point for the application. It manages the state and renders other components.
- PasswordGeneratorForm: A form component that allows users to input their password preferences such as length and character types.
- PasswordStrengthIndicator: A component that visually represents the strength of the generated password based on certain criteria.
- PasswordDisplay: A component that displays the generated password and provides a copy-to-clipboard functionality.

User Interaction:

- Users interact with the PasswordGeneratorForm component to input their preferences.
- The form's input fields and checkboxes capture the user's desired password length and character types.
- On form submission, the App component processes the input and generates a password based on the selected preferences.
- The generated password is displayed in the PasswordDisplay component, along with a copy-to-clipboard button.

Password Generation Logic:

- The App component utilizes the input values to generate a password that meets the specified criteria.
- It uses JavaScript functions to randomly select characters from the available character sets (uppercase, lowercase, numbers, and special characters) based on the user's preferences.
- The generated password is stored in the App component's state.

Password Strength Indicator:

- The PasswordStrengthIndicator component evaluates the generated password's strength based on criteria such as length, character complexity, and variety.
- It visually represents the password's strength using color-coded indicators, such as weak, moderate, or strong.

Copy-to-Clipboard Functionality:

- The PasswordDisplay component provides a button to copy the generated password to the user's clipboard.
- On button click, the component uses the appropriate JavaScript API or library to copy the password to the clipboard.

Responsiveness:

- The components are designed to be responsive and adapt to different screen sizes and devices.

- CSS media queries and responsive design techniques are used to ensure a seamless user experience across various devices.

Error Handling:

- The App component performs input validation and displays appropriate error messages if the user inputs invalid or incomplete preferences.

1.2 Scope

The scope includes creating a web application that enables users to create strong passwords. The product will include an intuitive user interface that allows users to alter password specifications like length and character kinds. To assist users in determining the security level of the passwords produced, the programme will display signs of password strength. The project will also have a copy-to-clipboard feature for quick use of the created password. A smooth user experience on various devices is ensured by the application's responsive design. Additionally, the initiative intends to inform users of the value of secure passwords and password security recommended practises.

1.3 Constraints

Ensure compatibility with popular web browsers, implement security measures to protect user data, optimize performance for fast load times, and conform to accessibility guidelines to ensure equal access for users with disabilities.

2 Technical specifications

The React Password Generator is a web application that allows users to generate secure passwords. It requires a stable internet connection and meets the minimum requirements for running a modern web browser. Users should follow best practices for managing their passwords and maintaining overall online security. The application is used primarily by individuals and not intended for enterprise-level password management or integration with external systems. The user data entered into the React Password Generator is assumed to be accurate and valid. The React Password Generator does not store or transmit user passwords or any sensitive information to external servers or third-party services without explicit user consent.

The most important details in this text are the data collected and stored by the application. These data include user preferences for password generation, generated passwords, password strength assessment, user interaction data, error logging and reporting, localization and translations, and data encryption, storage, and access control measures. User preferences for password generation include options such as password length, character types, and the exclusion of ambiguous characters. Generated passwords are generated by users and stored for future reference or integration with password management tools. Password strength assessment includes metrics or indicators used to evaluate password complexity and security.

User interaction data includes button clicks, form submissions, and navigation patterns. Error logging and reporting include errors or exceptions that occur during its operation. Localization and translations may be required if multi-language support is implemented. Data collected and stored by the application should be handled securely and in compliance with relevant data protection and privacy regulations.

- User input for password generation (length, character types).
- Data for password strength assessment (complexity, uniqueness, resistance).
- Localization data for multi-language support.
- Password policy data based on industry standards.
- User preferences and customization data (excluded characters, preferences).
- Data for password history and management (encrypted passwords, metadata).
- Security data (encryption protocols, data protection).
- Testing data (test cases, results, performance issues).

3 Technology stack

3.1 Software Requirements

- React: JavaScript library for building user interfaces.
- JavaScript: A programming language for application logic and password generation.
- HTML: A markup language for structuring web pages.
- CSS: Styling language for visual design.
- Git: Version control system for code management.
- GitHub: Web-based platform for hosting Git repositories and collaboration.
- Visual Studio Code: Source code editor with JavaScript and React support.
- Browser Developer Tools: Built-in browser tools for debugging and inspection.
- Package Managers: npm or Yarn for dependency management.
- Deployment Platforms: Netlify, Vercel, and GitHub Pages for hosting the application

3.2 Hardware Requirements

- The React Password Generator, being a web-based application, does not have specific hardware requirements. However, it is expected to run smoothly on standard computing hardware that meets the following general requirements:
- Processor: A modern processor (such as Intel Core i3 or AMD Ryzen 3) or equivalent that can handle web application processing efficiently.
- Memory (RAM): At least 4GB of RAM or higher to ensure smooth performance, especially when dealing with multiple tabs or browser instances.
- Storage: Sufficient storage space is required to store the application files and any associated data. A few megabytes of storage should be sufficient for the React Password Generator.
- Network Connection: A stable internet connection is necessary for accessing the React Password Generator web application and generating passwords

4 Proposed Solution

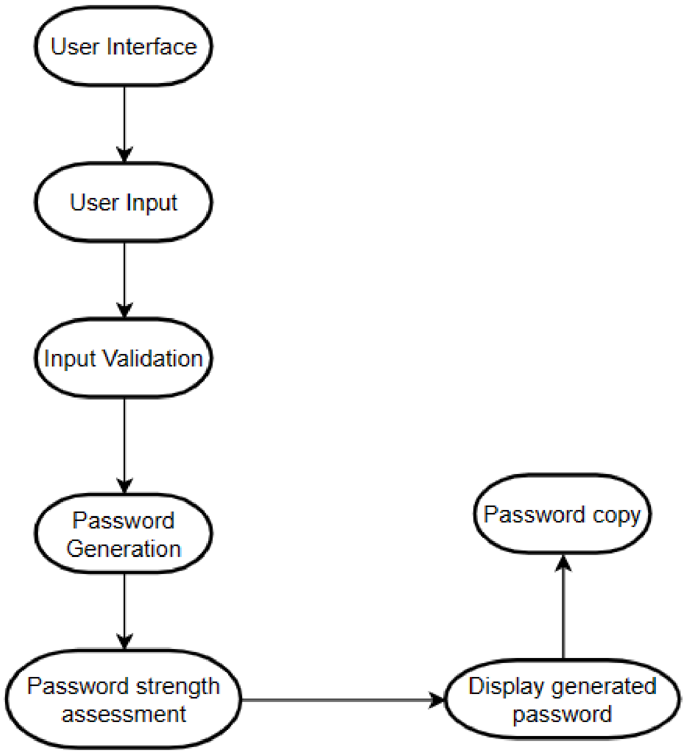
Password security is a pressing issue in the digital era, with data breaches and cyber threats on the rise. Existing password generators often lack customization options and intuitive interfaces, leading to weak passwords that compromise user accounts. The React Password Generator aims to address this problem by offering a standalone web application that empowers users to generate strong and secure passwords effortlessly.

The React Password Generator is a user-friendly web application designed to provide a customizable and secure solution for generating strong passwords. It offers a simple and intuitive interface, allowing users to specify password length and character types. It employs a robust algorithm to generate random and secure passwords that meet industry standards. It enables individuals to enhance their password security and protect sensitive information effectively.

5 User I/O workflow

A process flow is a visual representation that shows the sequence of steps or tasks in a process. It helps to understand, analyze, and improve workflows by illustrating the order and dependencies of actions and decisions. It is used in various industries for documenting and optimizing processes.

Proposed methodology



6 Exceptional scenarios

Step	Exception	Mitigation	Module
09-03-2023	1.1	First Draft	Sushil
18-03-2023	1.2	Added Workflow chart	Sushil

7 Key performance indicators (KPI)

- Password Strength Improvement: Measure the percentage of passwords generated by the application that meet or exceed industry-standard strength criteria. This KPI reflects the effectiveness of the password generation algorithm in creating strong and secure passwords.
- User Engagement: Track users engagement metrics such as the number of active users, session duration, and frequency of password generation. This KPI indicates the level of user adoption and satisfaction with the application.
- Error Rate: Monitor the occurrence of errors or failures during password generation, such as validation errors or technical issues. This KPI indicates the stability and reliability of the application.
- Accessibility Compliance: Assess the adherence of the application to accessibility standards, such as WCAG, by monitoring compliance metrics. This KPI ensures that the application is accessible to users with disabilities.
- Page Load Time: Measure the average time it takes for the application to load and be ready for user interaction. This KPI reflects the performance of the application and impacts user satisfaction.
- Security Audit: Conduct regular security audits to identify and address any vulnerabilities or potential security risks in the application. This KPI ensures the application's robustness in protecting user data.
- Customer Feedback and Ratings: Monitor user feedback, reviews, and ratings to gauge overall user satisfaction and identify areas for improvement. This KPI provides insights into user perception and helps prioritize future enhancements.