password manager

The password manager application is designed to provide users with a secure and convenient way to store, retrieve, and manage their sensitive password information across various websites. The application follows a structured methodology to ensure proper functionality and security measures.

The first step in utilizing the application is the user registration process, implemented in the `registration.php` script. This script allows new users to create an account by providing a unique username and a password. The script performs necessary validations, such as checking if the entered password and confirmed password match, and ensuring that the chosen username is not already taken in the database. If the provided information is valid, the script securely hashes the password using the SHA-256 algorithm and stores the username and hashed password in the `User_Login` table of the MySQL database. Upon successful registration, the user is redirected to the login page to authenticate with their newly created credentials.

The login functionality is implemented in the `login.php` script, which serves as the entry point for authorized users to access the password manager application. When a user submits the login form, the script establishes a connection with the MySQL database, prepares an SQL statement to retrieve the stored password hash for the given username, and compares it with the hashed version of the submitted password. If the credentials are valid, the user is logged in and redirected to the main dashboard page (`dashboard1.php`). If the authentication fails, an error message is displayed on the login form. This login system ensures that only authorized users can access and manage their sensitive password information within the application.

The `db_config.php` script plays a crucial role in centralizing the database connection logic. It defines database configuration settings as constants (database server, username, password, and database name) and provides a reusable function `getDbConnection()` to establish a connection to the MySQL database using those configuration details. This approach promotes code reusability and maintainability across different parts of the application that require database access, separating the concerns of connecting to the database from other application logic.

The main functionality of the password manager application is implemented in the `dashboard1.php` script, which serves as the central hub for authenticated users to manage their passwords. The script employs encryption techniques using OpenSSL to protect the stored passwords, ensuring their confidentiality. Users can save new passwords by providing the website URL and the password itself, which gets encrypted before storage in the database. Additionally, users can search for previously stored passwords by entering the website URL, and the corresponding decrypted password will be displayed. The dashboard interface provides a user-friendly way to manage passwords, allowing users to keep track of their credentials across multiple websites within the application.

$$E = L \times \ log2(R)$$

Password entropy is a measure of the randomness or unpredictability of a password. It quantifies the amount of information or uncertainty contained within a password, which directly relates to its strength against brute-force attacks or guessing attempts. Higher entropy means a password is more secure and harder to guess.

The entropy calculation considers two main factors:

1. **Password Length**: Longer passwords generally have higher entropy because there are more possible combinations.
2. **Character Set Size**: The range of characters used in the password (e.g., lowercase, uppercase, digits, symbols) also contributes to entropy. A larger character set means more possible combinations and higher entropy.

The formula for calculating password entropy is:

Entropy = Length × log2(Character Set Size)

For example, if a password is 8 characters long and uses a character set of 62 (uppercase, lowercase, and digits), the entropy would be:

Entropy = 8 × log2(62) ≈ 47 bits

The more entropy a password has, the stronger and more secure it is considered.

`entropy.php` implements a password strength checker for a password manager application. Its primary functionality is to assess the strength of a user-provided password based on its entropy, which is calculated by considering the length and

the character set used (uppercase, lowercase, digits, and special characters). The script categorizes passwords into three levels: weak, medium, and strong, based on predefined entropy thresholds. If a password is deemed weak or medium, the script suggests a randomly generated strong password for the user to consider. The purpose of this password strength checker is to educate users about password security best practices and encourage them to use strong, random passwords to better protect their sensitive data stored within the password manager application. By providing this tool, the application aims to promote the use of secure passwords, enhancing the overall security of the password management system for its users.

The overall purpose of the password manager application is to provide a centralized and secure solution for users to store, retrieve, and manage their sensitive password information, enhancing their online security and convenience. By implementing robust authentication mechanisms, encryption techniques, and a well-structured architecture, the application aims to offer a reliable and user-friendly experience for managing passwords across various online platforms.