**<file:///Users/jaypatel/Downloads/login_page%20(1).html-> website link**

**CodeVault: System Overview**

**Intended Use**

CodeVault (CV) is an online tool which is designed to help users securely organize, create, and store passwords for various websites. It is useful for individuals that are managing multiple passwords and teams requiring shared access to credentials. Users must create an account to establish their own password repository. To be able to add a new password, they must enter details such as the platform, username, and password. Additionally, CV features a password generator that will allow users to create strong passwords based on customizable specifications.

**Functionality Overview**

The system leverages **SQL** for database management and **Java** for password generation, among other functionalities. Development will take place in **Visual Studio Code**. When the user creates a password and associate it with a specific website, the system will store this information securely in the database. The database also maintains the master password and login credentials required for access. The user interface will be designed for ease of navigation, making sure an intuitive and organized experience that enhances overall user retention.

**Key Components**

1. **User Interface:**
   * It was developed using **React.js**, a JavaScript library known for building many dynamic and responsive interfaces.
   * **HTML5** and **CSS3** has been used for structuring and styling, making sure cross-device and cross-browser compatibility.
2. **Data Storage & Security:**
   * **PostgreSQL** will serve as the overall database, offering reliability and scalability.
   * Security measures will include encryption, access controls, and secret management to be able to protect sensitive information.
3. **Development Environment & Version Control:**
   * The project will be developed using **Visual Studio Code** with **Git** for the main version control.
   * The repository will follow a structured layout, making sure its organized source code, tests, documentation, configuration files, and assets.
4. **Security & Maintenance:**
   * The system will be built into **CI/CD pipelines** to automate testing, building, and deployment.
   * Issue tracking and project boards will facilitate and work through task management.
   * Code reviews will make sure quality before merging updates.
   * Regular backups, archiving, and monitoring will be able to help maintain long-term stability and security.

**Challenges Faced**

Making sure secure password storage required implementing encryption and hashing techniques like bcrypt. Designing a robust authentication system while being able to maintain usability was an overall challenge, especially in integrating multi-factor authentication (MFA). Optimizing database queries and structure was so necessary to maintain performance as the password repository grew. The frontend had to be both responsive and intuitive, which requires extensive testing across a bunch of different devices. Automating deployments with CI/CD pipelines led to debugging integration issues. Security threats like SQL injection, XSS, and brute-force attacks needed mitigation strategies like input validation and even rate limiting. Finally, being able to manage collaboration through Git required being able to maintain proper version control and handling merge conflicts efficiently.

**Networking Principles Used**

The **Domain Name System (DNS)** was really made to map the domain name to the hosting service’s IP address, making sure the website accessible. **IP addressing** makes sure that the server hosting the CodeVault system was reachable over the internet. **HTTPS encryption** was implemented using SSL/TLS to be able to havesecure data transmission and prevent eavesdropping. **Firewalls and access control lists (ACLs)** were set up to be able to protect the system from unauthorized access and network-based attacks. **Load balancing techniques** were considered to distribute traffic efficiently and be able to prevent server overload. **Protocols such as TCP/IP** makes sure reliable communication between the client, server, and database. **Network monitoring tools** were also utilized to be able to track system performance, detect potential threats, and even maintain uptime.