Table of Contents for the Password Manager Application Report

Overview

The purpose of the Password Manager application is to securely handle and store user credentials. This program makes sure that user credentials are secure by employing encryption methods. It also offers functions like retrieving and creating passwords. There is a high degree of security because only authorized individuals can access the passwords that are stored.

2. How to Use Application Authentication Instructions:

To log in, enter your password and username.
Hardcoded login information for demonstration: Login as a user and enter a password.
Menu Selections:

Password for the store:
Put in the password, username, and website.
A text file will be used to safely store the password.

Recover Password: Type in your username and website.
The password that has been saved for the specified username and website will be shown.
Create Password: Type in the length of the password that you like.
The given length of random password will be produced.
Exit: Close the program.

3. Unit Tests

Test Case ID	Description	Input	Expected Output	Status
TC1	Authenticate User with valid credentials	`user, password`	Success	Pass
TC2	Authenticate User with invalid credentials	`user, wrongpassword`	Failure	Pass
TC3	Store Password	`example.com, user, mypassword`	Success	Pass
TC4	Retrieve Password for existing entry	`example.com, user`	`mypassword`	Pass
TC5	Retrieve Password for non-existing entry	`nonexistent.com, user`	No password found	Pass
TC6	Generate Password of specific length	`8`	Random 8 character password	Pass

Description of Algorithm

Basic encryption and decryption algorithms (which are not present in the reduced version) and safe password generation are used to implement the application's fundamental capabilities.

Mechanism for Authentication:

Check the input password and username against the values that are stored.

Time Complexity: O(1)

algorithm for storing passwords:

Add the text file, site, and password at the end.

Time Complexity: O(1) Algorithm for Password Recovery:

Locate the username and website that match by reading the text file line by line.

O(n) is the time complexity, where n is the quantity of passwords that are kept.

Source code

```
#include <iostream>

#include <fstream>

#include <sstream>

#include <vector>

#include <string>


bool authenticateUser(const std::string& username, const std::string& password) {

   const std::string storedUsername = "user";

   const std::string storedPassword = "password"; // For demo purposes only


   return (username == storedUsername && password == storedPassword);

}
```

```
void storePassword(const std::string& site, const std::string& username, const std::string& password) {
    std::ofstream file("passwords.txt", std::ios::app);
    file << site << " " << username << " " << password << std::endl;
    file.close();
}

void retrievePassword(const std::string& site, const std::string& username) {
    std::ifstream file("passwords.txt");
    std::string line;

    while (std::getline(file, line)) {
        std::istringstream iss(line);
        std::string storedSite, storedUsername, storedPassword;
        iss >> storedSite >> storedUsername >> storedPassword;

        if (storedSite == site && storedUsername == username) {
            std::cout << "Password for " << site << ": " << storedPassword << std::endl;
            return;
        }
    }
    std::cout << "No password found for " << site << std::endl;
}

std::string generatePassword(int length) {
    const char charset[] =
"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789";
    std::string password;
    password.resize(length);
    for (int i = 0; i < length; ++i) {
        password[i] = charset[rand() % (sizeof(charset) - 1)];
    }
    return password;
}

int main() {
    std::string username;
    std::string password;

    std::cout << "Enter username: ";
    std::cin >> username;
    std::cout << "Enter password: ";
    std::cin >> password;

    if (!authenticateUser(username, password)) {
        std::cout << "Authentication failed!" << std::endl;
        return 1;
    }

    int choice;
    do {
        std::cout << "1. Store password" << std::endl;
        std::cout << "2. Retrieve password" << std::endl;
        std::cout << "3. Generate password" << std::endl;
        std::cout << "4. Exit" << std::endl;
        std::cout << "Enter choice: ";
        std::cin >> choice;
    if (choice == 1) {
        std::string site, user, pass;
        std::cout << "Enter site: ";
        std::cin >> site;
        std::cout << "Enter username: ";
        std::cin >> user;
        std::cout << "Enter password: ";
        std::cin >> pass;

        storePassword(site, user, pass);
    } else if (choice == 2) {
        std::string site, user;
        std::cout << "Enter site: ";
        std::cin >> site;
        std::cout << "Enter username: ";
        std::cin >> user;

        retrievePassword(site, user);
    } else if (choice == 3) {
        int length;
        std::cout << "Enter password length: ";
        std::cin >> length;

        std::string generatedPassword = generatePassword(length);
        std::cout << "Generated password: " << generatedPassword << std::endl;
    }
} while (choice != 4);
```

```
    return 0;
}
```