MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

NATIONAL TECHNICAL UNIVERSITY

"KHARKIV POLYTECHNICAL INSTITUTE"

Department of Computer Engineering and Programming

«Software Means of Information Protection »

*Laboratory work report No 1-2*

*Topic: «* Development of a user authority delimitation program based on password authentication*»*

Student:

Group. KH 919 i.e.

Ayoub EL-HADDADI

Verified by:

Lecturer Viktor CHELAK

Kharkiv – 2022

***Purpose of work:***

To study the principles of organization of password protection of programs, acquaintance with the types of passwords, implementation of password protection.

***Individual task:***

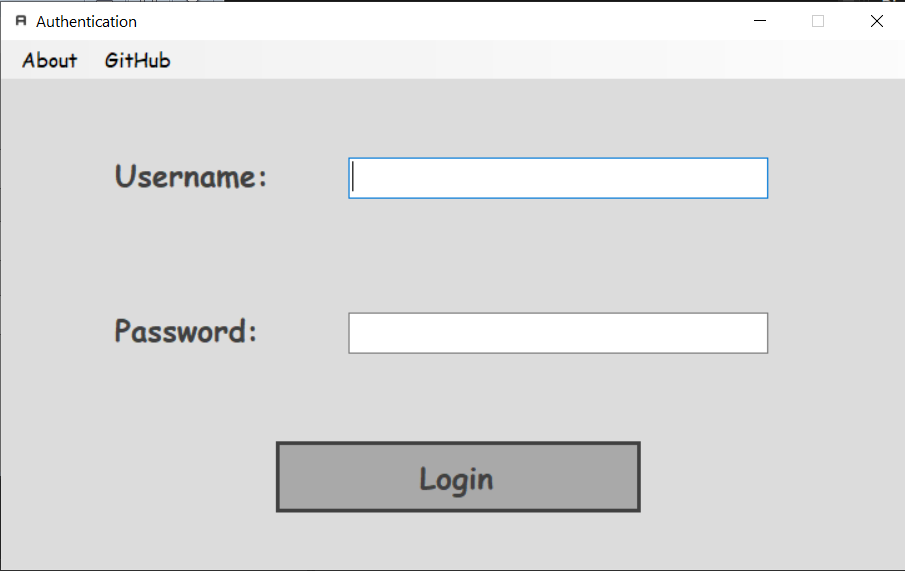
Variant 8:

Organization of password protection: *Password encryption*.

Restrictions on selected passwords: *The presence of Latin letters and symbols of the Cyrillic alphabet.*

***Algorithm of the program:***

*#Form1: (Authentication)*



*Figure 1 – Login form*

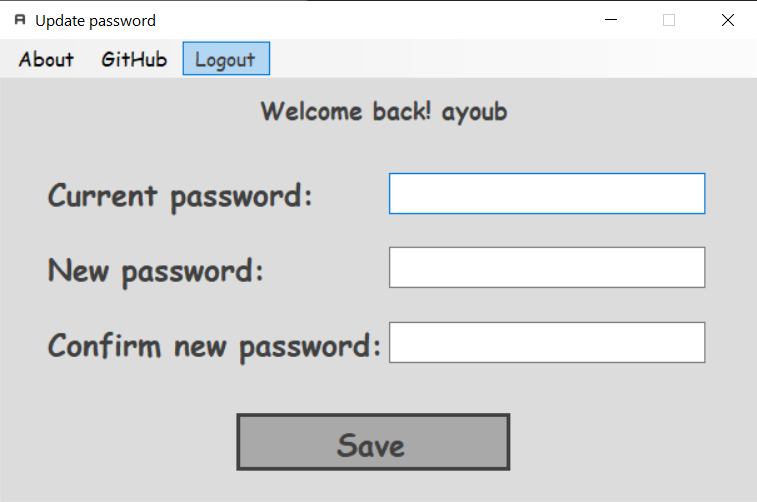
This is the first Form of the program, which we use to authenticate the user (or admin) depending on users existing in “**users.json**” file. And, if the file doesn’t exist the program creates a new file including one admin with blank password.

Checking the file if exist or to create new one if not we use on load event. And, for authentication and login to the program, we use on click event for the login button.

Passwords in “**users.json**” file are saved in encrypted form, for that I’ve created, another class “**PasswordHasher.cs**” (to avoid duplicates) with two methods, one for hashing password, and another for verifying.

After checking and verifying inputted data, we procced to next form. And, next form depends on user role (user, admin).

*# User mode (If authenticated user is not an admin):*



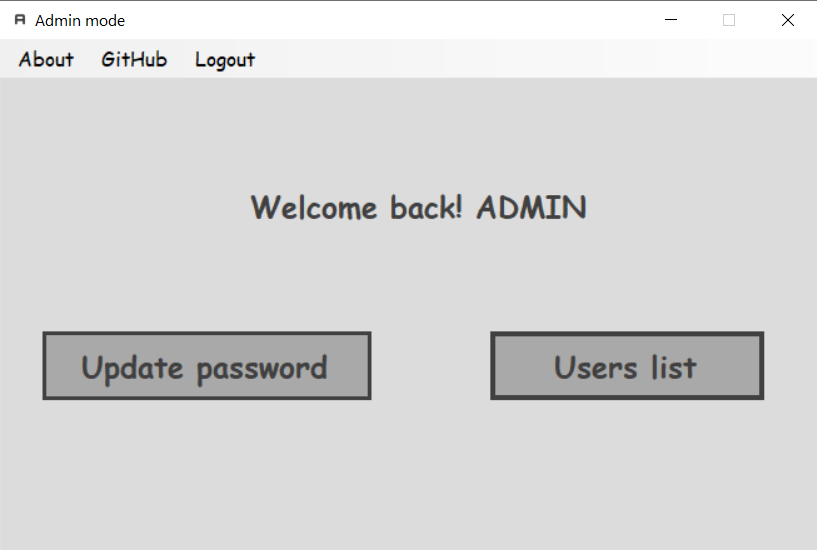
*Figure 2 – Update password form*

This form support only the functions of changing the user's password (if you enter the old password correctly) and going back to authentication form.

First step in click event on save button is to check current password, if the authenticated user is new (new users have blank passwords), we must leave that field blank and proceed to next step, if not we must verify this current password with **“VerifyPassword”** method in PasswordHasher class. If current password is not correct we popup a message box with error message.

Then, if current password is correct, we compare new password and new password confirmation, if not the same we popup a message box with error message. And, if it’s confirmed, we check this password with restriction given (edit some characters if needed) and hash this password and update it in “**users.json**” file, and automatically logging out.

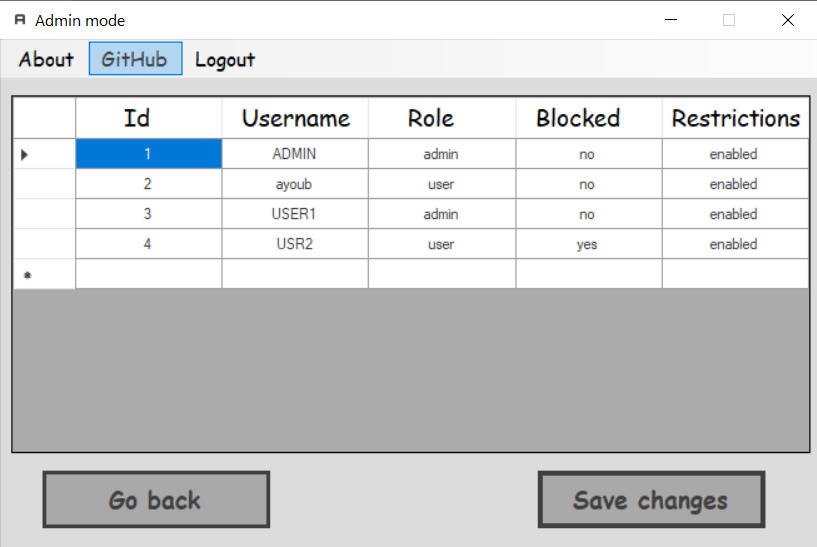
*# Admin mode (If authenticated user is an admin):*



*Figure 3 – Admin mode form*

This form support updating password like the one in user mode. So, in click event of the update password button we just redirect user to the user mode form.

Also, this form support a list of users, and ability to edit all fields except password. For displaying of this data we just need to read from “**users.json**”, and display it using data grid view in form of table, and enable updating/deleting in properties.



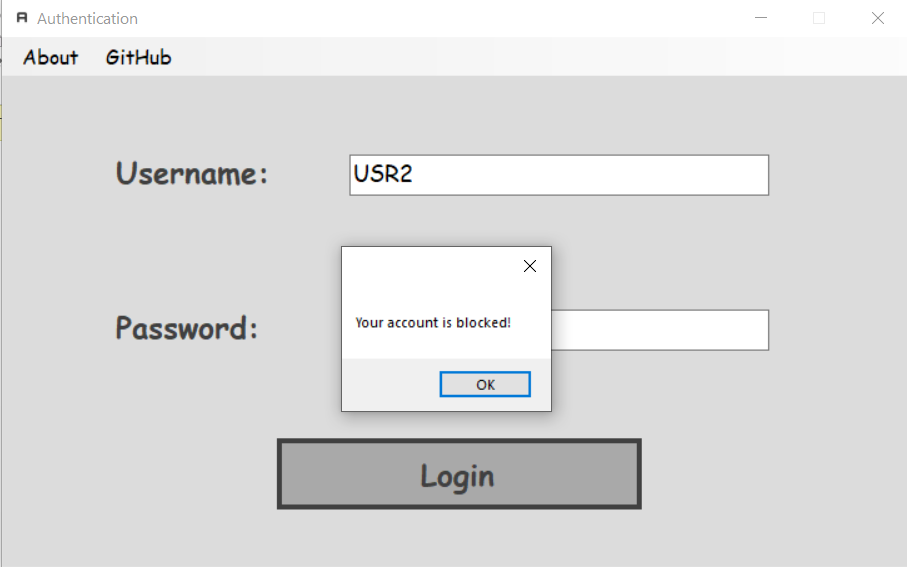
*Figure 4 – Users list*

The following is the users.json file for this list:



*Figure 5 – users.json*

The admin can block the user or disable password restriction for any user in this list, after updating or/and deleting data you must click “Save changes” button to save the changes, the click event for this button is to read from this data grid view and rewrite the users.json file.



*Figure 6 – Login with blocked user*

**Source Code:**

[*https://github.com/Elh-Ayoub/SMoIP\_Labs/tree/main/Lab1*](https://github.com/Elh-Ayoub/SMoIP_Labs/tree/main/Lab1)

**Conclusions:**

For this laboratory work, I have gained principles of organization of password protection of programs, acquaintance with the password encryption for password protection.