Password Manager

Description

In this assignment, you will create a program in Java that is a password manager. The GUI part of the project is being provided to you, you just need to implement the encryption, storing of passwords, removing passwords, etc. For the GUI, you should leave it as is so it is clear how to grade it, but you can customize the colors and themes if you would like. You should use the Java cryptography examples in class to aid in completing this assignment. I highly encourage you to break apart components of the examples into separate methods (e.g. encrypt, decrypt, hash, etc) instead of having one very long method (e.g. main).

Your password manager should allow you to do the following:

1. If no password file exists, take the initial password that is provided to create a key (use PBKDF2). You should then store the salt along with a token that you can use to verify in the future that the user entered the correct password. The token can be a simple word like "Cleveland" that you encrypt using the salt you store and the key generated from the password the user enters to verify that you can successfully decrypt it. This is how you can determine if the user has the correct password.
2. If a password file exists, use the password they enter along with the stored salt to verify the user entered the correct password by decrypting the token that you stored in Step 1. The same salt will be used for every password.
3. Allow the user to do the following through the UI:
    1. Allow them to add a password by taking the form fields and creating a new password and storing it in the file.
    2. Allow the user to delete a password by ensuring the file with the passwords does not contain the new one and removing it from the model.
    3. Allow the user to update a password by writing the new label and password to the file.
4. The file should be stored in the default directory of the program and should have the following format (the space between each is a tab character, which is already in the template provided)

```
salt encrypted_token
label1    encrypted_password1
label2    encrypted_password2
```

Below is an example of the file from a solution that was created:

```
xvbKAm57v61oYaftlj35Lg== GqlR9TQey6MvyuQPZqvTDg==
Canvas    Cxn9qggt+g8KWUwkh7Nhs0l0na/xFWAnuu7kT4HVd8U=
SIS  +6GGBGEzIBciWI9a+URBemhIPn29GE/uxwVWoSdbDAo=
```

A demo of the application will be provided in class, which you can use as a reference.

Grading

You will be graded as follows

- 10pts - Program meets all requirements above where passwords can be added, read from, and deleted. Program successfully checks if a file exists and if the correct password is used for an existing file.
- -3pts - Program does not successfully check if the existing password is correct, but all other functionality works if you start with a new file and new password.
- 0pts - Program has a compilation error, crashes, does not encrypt the passwords in the file, or does not meet the grading criteria above.

Submission

You can work on this assignment in groups of 2-3. If you do work in a group, only one person from the group submits the assignment. Please include names of the students you partnered with on this assignment so that they can receive credit. These names should be included as comments to Canvas.

Submit a complete zip of your project from the top directory.