In the essay submitted in the unit 9 of the 'Launching into Cyber Security' module, a few spoofing threats were identified applicable to the appointment and scheduling management information system (ASMIS). These threats included taking over an existing user’s account and upgrading a lesser privileged account to one with more privileges.

For this reason, a program was written in Python (Downey, 2009) which enforces strong password practices, applies cryptographic hashing functions to protect this privileged data, locks the user accounts after too many incorrect login attempts and separates patients’ data from the more privileged staff data.

The program enforces the following password restrictions:

* The password length must be between 8 and 64 characters. The 256-bit hashing function the program uses allows up to 64 hexadecimal characters (PythonPool, 2021).
* The password must contain special characters, numbers and upper-case letters.
* The password must contain at least one whitespace. Passwords with white spaces tend to be more secure (InfosecMatter, 2021).

The code will also require users to confirm their password to help them catch any typos.

Passwords which meet all the requirements will then be hashed using the SHA-256 cryptographic hashing method and stored in the csv file together with the user’s email address, registration date and the number of login attempts (set to 3 by default for new users).

To log in to ASMIS, the users will have to enter their email address and password. The entered password will be hashed and compared to the one in the database. If the passwords match, the user will be allowed to login.

The login process was also designed to not reveal whether the user is registered or not, i.e. the login function will ask for both email address and password even if the user does not exist in the csv file.

Code execution instructions:

1. Run the program to view three menu options in the terminal: login, register and exit.
2. Option can be selected by typing it inside the terminal window and pressing ‘Enter’ on your keyboard.
3. If the register option is selected:
   1. The patient must enter their email address.
   2. The patient must enter their password which is longer than 8 characters, less than 65 characters, contains upper-case letters, special characters and whitespaces.
   3. If the password meets the criteria, the user will be asked to repeat the password.
   4. If the two passwords match, the user’s account will be created, and the user will be taken back to the menu where they can choose one of the three options again.
   5. If the user’s account already exists, they will be notified when trying to register and taken to the login page instead.
4. If the login option is selected:
   1. The patient must enter their email address and password.
   2. If the email address belongs to an existing user and the password matches the hashed password stored in the database, the user is logged in and the program exits.
   3. If the email address belongs to a person who hasn’t yet registered, the user is asked to enter their password regardless. The user is then presented with an option to either continue with the login by inputting their email address or typing ‘exit’ to quit the application.
   4. If the email belongs to an existing user but the password is incorrect, the user is only given 3 login attempts to access their account. When the number of login attempts is exceeded, the user’s account is locked, and the user is asked to contact the Reception.
5. If the exit option is selected, the program exits.

**Testing summary**

The program was run in Codio and it was confirmed that the menu was displayed in the terminal with the following three options as expected: Login, Register and Exit. The ‘Patients.csv’ file was also correctly created upon the initial run.

A screenshot of a computer

Description automatically generated with medium confidence

The Patient’s file was confirmed to have been populated with 4 columns: email, password, registered\_on and login\_attempts. The file was also confirmed to be empty except for the column titles.

Graphical user interface, text, application

Description automatically generated

Using the terminal, an invalid menu option was input and the Enter button was pressed on the keyboard to confirm the user is asked to input a valid selection.

Text

Description automatically generated

It was then tried to log in and it was confirmed that this was not allowed, since there are no registered users saved in the csv file yet.

User tried to log in a few more times, confirming each time this was not allowed.

Text

Description automatically generated

The ‘menu’ option was then typed into the terminal to confirm the user is correct returned to the three menu options.

Text

Description automatically generated

Register option was typed into the terminal next to confirm the user is asked for an email address to start with. All upper-case letters were used to ensure the conversion to the lower case in the code is working correctly.

Graphical user interface, text

Description automatically generated

The email address restrictions were then tested, and it was confirmed that if the email did not contain one @ symbol, at least one dot and no spaces, the registration was not allowed to proceed to the next step.

Text

Description automatically generated

If a valid email address was provided, the user was allowed to enter their password.



The passwords restrictions were tested next confirming appropriate error message is presented otherwise.

1. The password must be at least 8 characters long.
2. The password must not be more than 64 characters long.
3. The password must contain at least one special character.
4. The password must contain at least one upper case letter.
5. The password must contain at least one number.
6. The password must contain at least one whitespace.

Text

Description automatically generated

A password meeting the requirements was input next, confirming this correctly triggers the password confirmation request.

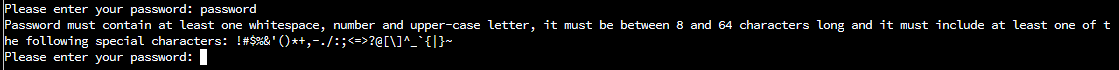


A password different to the initial was then entered to confirm the user is asked to re-enter the passwords.

Text

Description automatically generated

A password which doesn’t match the criteria was entered again to confirm that recurring function correctly triggers the password suitability checks.

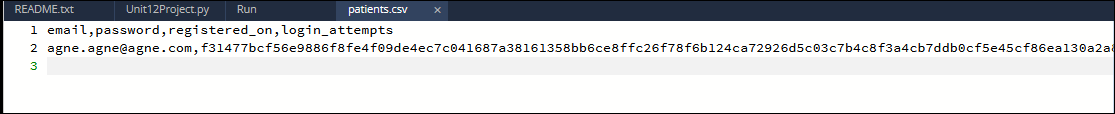


A suitable password was provided next followed by entered a password which matches the initial password and then it was confirmed that the user is correctly notified upon the successful registration and taken back to the main menu.

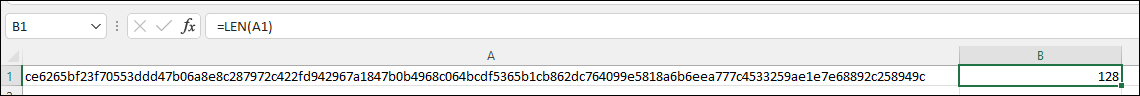
Text

Description automatically generated

The Patients csv file was then opened, and it was confirmed that the user was inserted into the file. The user’s email was confirmed to remain unchanged, the registration date was confirmed to be correct and the login\_attempts were confirmed to be correctly set to 3 for a new user.



It was also confirmed that the password was hashed to the correct length of 128 characters. SHA256 hash function is 256 bits (32 bytes) long which is equal to 64 hexadecimal characters (Anderson, 2020). The salt is 32 bytes long, so the total hashed password length must be 126 hexadecimal characters. The password length was verified in Excel using the LEN function.



A few more users were registered, and they were all stored in the Patients csv file as expected.

Graphical user interface, application

Description automatically generated

It was then tried to log in as one of these registered users confirming it is permitted. The password entered into the terminal should be hashed and compared to the one in the csv file. If the correct password was entered, the two should match and the user should be able to log in.

During the login upper case letters were used for the email to confirm it was correctly converted to the lower case before being compared to the email address value used in the database.

Text

Description automatically generated

The program was run again and this time it was tried to log in with the registered user’s email address but an incorrect password. It was confirmed that this was not allowed, that the remaining login attempts were displayed (these should be 2) and that the login attempts were updated in the csv file to 2 as well.

Text

Description automatically generated

To update the login attempts, a new temporary csv file had to be created whilst the old one had to be renamed. The old file was then deleted, and the temporary file was renamed to Patients. It was confirmed that all three users still existed in the new Patients file.

Word

Description automatically generated with medium confidence

It was then attempted to login as the same person with an incorrect password two more times and it was confirmed that an ‘Account locked’ message was triggered and the program was exited. The login attempts count was also set to 0 in the file.

Text

Description automatically generated

The program was started up again and it was tried to log in as the locked out person, it was confirmed that it was still not allowed and the message was displayed as soon as the email address was entered.

Text

Description automatically generated

It was also confirmed that if the user uses capital letters in their email address when registering, these are converted to lower case letters before being stored in the csv file.

Graphical user interface

Description automatically generated

Lastly, the ‘exit’ menu option was used, and it was confirmed that the program was exited.

Text

Description automatically generated

The Patient csv file was then deleted, and the program was run twice (exiting each time), confirming that the file size check worked correctly, and headers were not written to the file each time the program was run.

A screenshot of a computer

Description automatically generated with medium confidence

**References**

PythonPool. (2021) Python SHA256: Implementation and Explanation. Available from: <https://www.pythonpool.com/python-sha256/> [Accessed 6 February 2022].

InfosecMatter. (2021) Spaces in Passwords – Good or a Bad Idea? Available from: <https://www.infosecmatter.com/spaces-in-passwords-good-or-a-bad-idea/> [Accessed 5 February 2021].

Anderson, R. (2020) *Security Engineering: A Guide to Building Dependable Distributed Systems*. 3rd ed. Indianapolis, Indiana: John Wiley & Sons.

Downey, A., Elkner, J. & Meyers, C. (2012) How to Think Like a Computer Scientist. Learning with Python. Massachusetts, USA: Green Tea Press.

Prajapati, D. (2021) Hashing Passwords With pbkdf2\_hmac. Available from: https://cppsecrets.com/users/144671001011011125755485053504851504964103109971051084699111109/Python-Hashlib-HASHING-PASSWORDS-WITH-PBKDF2HMAC.php [Accessed 4 February 2021].