**Documentation for my Y11 T2 Task**

**Planning documentation**

**Task definition**

I have been given the challenge of designing a Python application to address the common cybersecurity issue of weak passwords. My application, named “Password Strength Checker”, Is designed to generate passwords, as well as how strong it is, whilst also checking your submitted password against breached passwords and 100k common passwords, determining your password's strength with an estimate in years.

This program has 4 core features, which are as follows:

Customisable generation settings: You are able to set all these settings in an extra window to determine what your generated password will be made up of. You can also change the length to a maximum of 16 characters.

A blue screen shot of a list of words

AI-generated content may be incorrect.

You can copy your selected password to the clipboard, saving time and improving user experience.

You can hide your password, allowing for that extra security when generating or checking your password.

Your password is compared to passwords that have been found in data breaches and English words to ensure its long-lasting security.

**Algorithm Design for “generate password” function**

This is the pseudocode algorithm design for my generate password logic, probably the most complex algorithm in my program

FUNCTION generate\_password(settings, length):

SET chars TO lowercase letters (a-z)

IF settings["use\_capital\_letters"] is True:

ADD uppercase letters (A-Z) to chars

IF settings["use\_numbers"] is True:

ADD digits (0-9) to chars

IF settings["use\_special\_characters"] is True:

ADD special characters (!@#$%^&\*()-\_=+[]{}|;:,.<>/?) to chars

IF chars is empty:

SET chars TO all ASCII letters (a-zA-Z)

password = ""

FOR i FROM 1 TO length:

ADD random character from chars to password

RETURN password

**Program Storyboard**

This storyboard presents my preliminary design of the theorised user interface, using ttk and tk respectively on a python 3.x framework. This application features input fields for various data entries alongside various setting pages to configure generation settings. It also contains a copy to clipboard button and a hide and show password button to improve the user experience. My envisioned program also contains an about me section to summarise what the program's purpose is, along with crediting its creator.

A screenshot of a computer program

AI-generated content may be incorrect.

**Implementation**

**Git repository Link:** [**https://github.com/RomanBleyer/term\_2\_Y11\_password\_strength\_checker/tree/main**](https://github.com/RomanBleyer/term_2_Y11_password_strength_checker/tree/main)

**Testing:**

**Project test table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Variable name** | **variable data type (string, dictionary, etc)** | **Expected output** | **Description of variables purpose** |
| **DICTIONARY** | |  | | --- | | **set[str]** |  |  | | --- | |  | | |  | | --- | | **Words (≥4 characters) from nltk.words()** |  |  | | --- | |  | | **Used to detect words inside a password** |
| **COMMON\_PASSWORDS** | |  | | --- | | **set[str]** | | **A set of the 100k most common passwords** | **To check if the entered password is a common password** |
| **widgets and settings\_widgets** | |  | | --- | | **list[tk.Widget]** |  |  | | --- | |  | | **List of active widgets to allow for page swapping** | **Clears active widgets before loading a page** |
| **entropy** | **Float** | **Calculated from len(password) \* log2(charset\_size)** | **Used to estimate guessing difficulty** |
| **what\_page\_am\_i\_on** | **str** | |  | | --- | |  |  |  | | --- | | **"main\_menu" / "password\_generator\_**  **settings\_menu" / “about\_me\_and\_**  **help\_menu”** | | **Used to determine what widgets to place when switching pages** |

**Evaluation**

**Project Reflection**

Developing this Program was a challenging yet rewarding experience. This project combined key elements of user interface design, security best practices, and Python programming into a cohesive tool. I learned how to use libraries like tkinter, Pillow, and nltk effectively, while also integrating external services like pwnedpasswords for real-world password breach detection.

Overall, this project strengthened both my technical and design skills, and I now have a greater understanding of how software can guide users toward better digital safety habits.