

Python Password Strength Checker Project

...

Altolane Jackson (AJ)

Password Strength Checker (Python CLI Project)

Project Type: Python Terminal-Based App

- helps users evaluate their password security and generate strong passwords.
- It runs in the terminal and follows key software development principles.

Why this project?

- Strong passwords are a crucial part of cybersecurity.
- This tool helps educate users about what makes a password strong.
- It also shows how simple logic and Python structures can create useful tools.

```
=== Password Strength Checker ===  
1. Check password strength  
2. Generate strong password  
3. Exit  
Select an option (1-3): █
```

Features Overview

- **There's an interactive menu: check a password, generate one, or exit.**
- **If a password is weak, the app gives improvement suggestions.**
- **It uses a custom scoring system based on length and character variety.**

```
=== Password Strength Checker ===  
1. Check password strength  
2. Generate strong password  
3. Exit  
Select an option (1-3): 2
```

```
Select an option (1-3): 1  
Enter your password: fycug
```

```
Strength Score: 1 / 6
```

```
Feedback: ❌ Weak Password
```

```
Suggestions to improve:
```

- Password is too short. Use at least 8 characters.
- Add uppercase letters.
- Include numbers.
- Include special characters (e.g. !@#\$%).

```
Strength Score: 3 / 6
```

```
Feedback: ⚠️ Moderate Password
```

```
Suggestions to improve:
```

- Add lowercase letters.
- Include special characters (e.g. !@#\$%).

```
Enter your password: AJwnvvownv234567!@#$
```

```
Strength Score: 6 / 6
```

```
Enter your password: Aj12345!@#
```

```
Strength Score: 5 / 6
```

```
Feedback: ✅ Strong Password
```

Programming Requirements

- **Descriptive Variable Names:**
Used clear, meaningful names like `user_password`, `strength_score`, `feedback_messages`
- **Three Data Types:**
Included strings (password input), integers (score values), and booleans (character type checks)*
- **Decision-Making Structures:**
Used `if`, `elif`, and `else` to score strength and guide user interaction

```
password_strength_checker.py x
1  # Password Strength Checker CLI Tool
2  # Author: AJ
3  # Purpose: Evaluates password strength and gives improvement suggestions using custom logic
4
5  import string
6  import random
7
8  def evaluate_password_strength(user_password):
9      """
10     Evaluates the strength of a given password based on length, character types, and uniqueness.
11     Returns a tuple: (strength_score, feedback_messages).
12     """
13     strength_score = 0
14     feedback_messages = []
15
16     # Booleans to track character variety
17     has_lowercase = any(char.islower() for char in user_password)
18     has_uppercase = any(char.isupper() for char in user_password)
19     has_digit = any(char.isdigit() for char in user_password)
20     has_symbol = any(char in string.punctuation for char in user_password)
21
22     # Evaluate length
23     if len(user_password) >= 12:
24         strength_score += 2
25     elif len(user_password) >= 8:
26         strength_score += 1
27     else:
28         feedback_messages.append("Password is too short. Use at least 8 characters.")
29
30     if has_lowercase:
31         strength_score += 1
32     else:
33         feedback_messages.append("Add lowercase letters.")
34
```

- **Loops for Repeated Tasks:**
while True loop enables continuous user access without restarting the program
- **Reusable Functions:**
Defined `evaluate_password_strength()` and `generate_strong_password()` for clean, modular logic
- **List (Sequence) Use:**
Stored improvement tips in `feedback_messages[]` and iterated through them with a for loop
- **Code Documentation:**
Docstrings and inline comments throughout explain each function's purpose and logic