ISHA
A50105223112

# MINOR PROJECT REPORT

**Project title**: Password Strength Checker using Python

**Submitted by:** ISHA

# 1. Introduction

In the digital era, users access many online platforms that store sensitive personal information. Password based authentication is a standard practice, but many users tend to choose weak passwords. Weak passwords make systems vulnerable to brute-force attacks, data leaks and hacking. Hence there is a need for a tool that checks password strength and prevents users from creating insecure passwords.

This project focuses on creating a simple yet effective GUI based **Password Strength Checker** built in Python using the Tkinter library. The tool analyzes the strength of user-entered passwords based on entropy, character diversity (uppercase, lowercase, digits, symbols), and pattern validation using Regular Expressions (Regex).  A Password Strength Checker helps ensure users create secure passwords that are difficult for attackers to guess or crack. It reduces the risk of unauthorized access and data breaches.

It's basic features are as following:
- Checks whether a password is strong, moderate, or weak.
- Encourages the use of secure and complex passwords.
- Applying entropy calculation to estimate password unpredictability.
- To give feedback using a GUI interface.

The project is designed for personal and educational use. It checks passwords based on
Five key rules:

- Minimum length of 8 characters
- At least one uppercase letter
- At least one lowercase letter

- At least one digit
- At least one special character

---

# 2. Literature Review

From Very Weak to Very Strong: Analyzing Password-Strength Meters : In this article[1], the author looks at password strength checkers used on popular websites. These tools are supposed to help users create stronger passwords. While some do a good job, many are not designed properly and don't explain how they work. The researchers tested these checkers using common passwords and found that the same password often got different ratings on different websites. Sometimes, even weak passwords were marked as "strong." This can confuse people and lead them to use weak passwords by mistake. The study shows that these tools need improvement so they can truly help users stay safe online.

Analyzing password strength: In this study[2], the author studies that Passwords are still the most common way to log in because they're easy to use and don't need special tools. But many people choose weak passwords that are easy to guess. Even though websites use rules (like needing capital letters or symbols) and password strength meters, over 90% of real passwords are still unsafe. This is because users often want something easy to remember, not necessarily strong. The paper studies a large set of real passwords and creates a better password checker to help users choose stronger and safer passwords.

Monte Carlo Strength Evaluation: Fast and Reliable Password Checking : This paper[3] introduces a new method to estimate how many guesses an attacker would need using these advanced techniques. This method is fast, doesn't need a lot of computer power, and works with many types of attack models. It also gives reliable results.

Real Time Password Strength Analysis on a Web Application Using Multiple Machine Learning Approaches: This study[4] introduces a new system that uses machine learning (like Decision Trees, Naïve Bayes, and Neural Networks) in a web application to check if a user's password is strong. The system only allows login if all the models agree the password is strong. Testing showed Decision Tree was the most accurate (99%) They created 250 test accounts — 150 with strong passwords and 100 with weak ones. Attackers couldn't crack any of the strong passwords, but they cracked 86 of the weak ones. This shows that the system helps force users to create safer passwords and protects against common attacks.

ISHA
A50105223112

Passwords and Python: Introducing Security Concepts in Lower-Division Programming: This assignment[5] is designed for beginner Python students to help them learn about password strength and security. Students build a password strength checker and management system using basic programming concepts like conditionals, loops, and logical operators. It also includes a slide deck to teach security basics and password concepts, along with discussion questions to encourage deeper thinking about online security.

Graphical password management system project: This study[6], explains that Graphical passwords use drawings instead of typed passwords. This study explores using finger-drawn doodles or simple signatures for login. It checks both the drawing and the way it's drawn (speed, movement) for added security. While harder to fake, these passwords are easy to forget and can be drawn differently each time, which can cause problems.

A Large-Scale Evaluation of High-Impact Password Strength Meters: In this study [7] , the author explains thatPassword meters are tools that help users create strong passwords, but many are poorly designed and inconsistent. This study analyzes popular meters and finds that the same password can be rated very differently across sites—sometimes calling weak passwords "strong." These issues can confuse users, but the research offers ways to improve these tools over time.

Analyzing Password Strength: A Combinatorial Entropy Approach: This study[8], checks how strong passwords are and how easy they are to guess. It compares different ways to measure password strength and introduces a new method using **combinatorial entropy** to make passwords harder to crack. The results help improve password security and safer login systems.

---

# 3. Methodology

## Entropy Calculation

In Cybersecurity, entropy is a mathematical way of measuring how hard it is to guess a password. The more unpredictable password is, the stronger and more secure it is.
A password with More characters, More variety (letters, numbers, symbols) and less predictable patterns will have higher entropy and be much harder to crack.
Entropy can be calculated using following formula:

ISHA
A50105223112

## Length and Complexity Check

The program ensures that the password meets the minimum requirements in terms of    length and includes various character types.

## Regex Validation

Regex- Regular Expressions are used to validate the structure of the password by checking for the presence of uppercase, lowercase, number and special characters.

## GUI Development

A GUI (Graphical User Interface) is a visual interface that allows users to interact with a program using buttons, text fields, and labels instead of typing commands in a terminal .
For this project, the Tkinter library is used. Tkinter is the standard GUI toolkit for python.
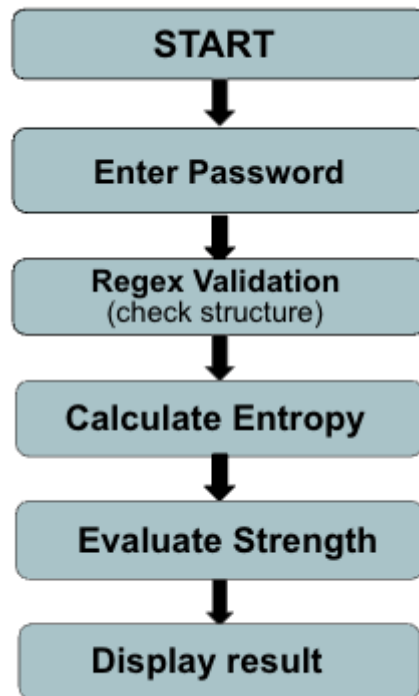
# 4. Flowchart and diagrams



Figure 1 depicts the working of password strength checker

# 5. Implementation

```python
import tkinter as tk
import re

# Function to check how strong the password is
def check_password():
    password = entry.get()
    strength = 0
```

```python
    if len(password) >= 8:
        strength += 1
    if re.search(r"[A-Z]", password):
        strength += 1
    if re.search(r"[a-z]", password):
        strength += 1
    if re.search(r"\d", password):
        strength += 1
    if re.search(r"[!@#$%^&*]", password):
        strength += 1

    if strength <= 2:
        result_label.config(text="Weak Password")
    elif strength == 3 or strength == 4:
        result_label.config(text="Moderate Password")
    else:
        result_label.config(text="Strong Password")

# Interface Design
root = tk.Tk()
root.title("Password Checker")

tk.Label(root, text="Enter your password:").pack()
entry = tk.Entry(root, width=25)
entry.pack()

tk.Button(root, text="Check", command=check_password).pack()

result_label = tk.Label(root, text="", font=("Arial", 12))
result_label.pack()

root.mainloop()
```
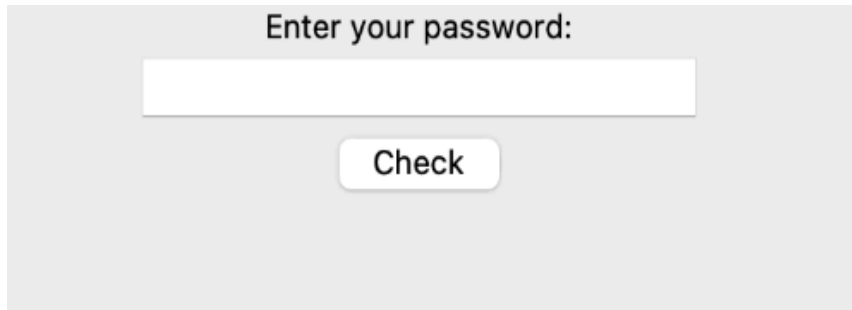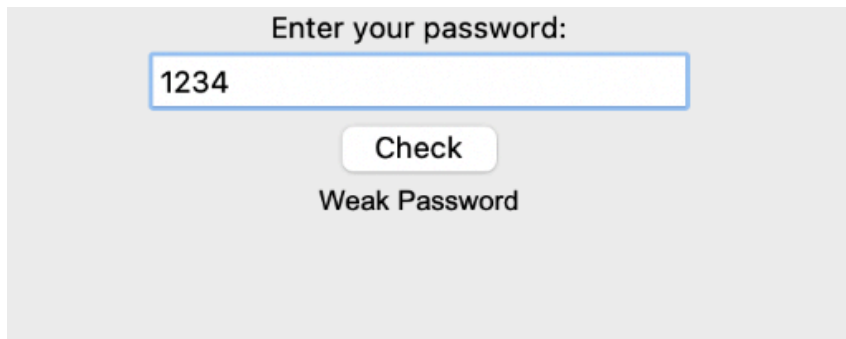
Figure 2 shows the main window of the password strength checker.



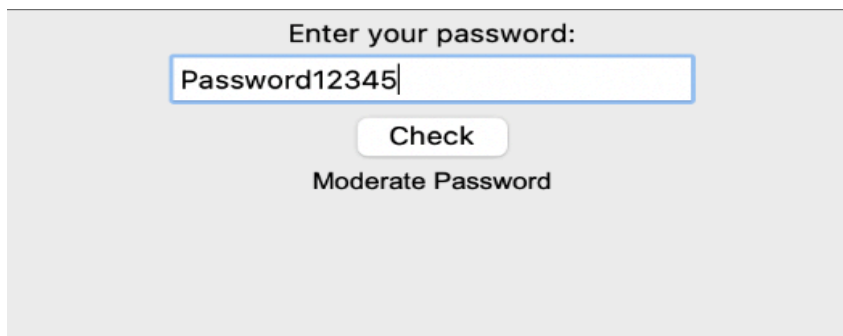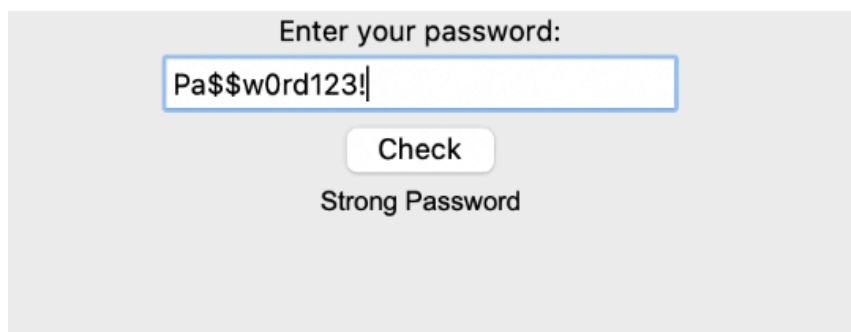Figure 3 shows a weak password because it does not include any uppercase, special characters or symbols.



Figure 4 shows a moderate password because it does not include any special character or symbol.

Figure 5 shows a strong password including a special character, an uppercase, lowercase and numbers.

---

# 5. Conclusion

In this project, a **Password Strength Checker** was successfully developed using Python and Tkinter. The application allows users to input a password and instantly receive feedback on its strength — categorized as Weak, Moderate, or Strong — based on five commonly accepted criteria:

- Minimum length of 8 characters

- Use of uppercase letters

- Use of lowercase letters

- Inclusion of numbers

- Use of special symbols (like @, #, etc.)

This project helped in understanding basic Python programming, GUI development, and the use of regular expressions to analyze user input. It also emphasizes the importance of creating strong and secure passwords to protect user data in real-world applications.

# References

1. De Carnavalet, X.D.C. and Mannan, M., 2014, February. From Very Weak to Very Strong: Analyzing Password-Strength Meters. In *NDSS* (Vol. 14, pp. 23-26).
2. Devillers, M.M., 2010. Analyzing password strength. *Radboud University Nijmegen, Tech. Rep*, *2*(10).
3. Dell'Amico, M. and Filippone, M., 2015, October. Monte Carlo strength evaluation: Fast and reliable password checking. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security* (pp. 158-169).
4. Farooq, U., 2020. Real time password strength analysis on a web application using multiple machine learning approaches. *Int J Eng Res Technol (IJERT)*, *9*(12), pp.359-364.

ISHA
A50105223112

5. Fiesler, C., Dalal, S. and Paup, J., 2023. Passwords and python: introducing security concepts in lower-division programming.

6. Acharya, K., 2021. Graphical password management system project report. *Available at SSRN 4943460*.

7. Carnavalet, X.D.C.D. and Mannan, M., 2015. A large-scale evaluation of high-impact password strength meters. *ACM Transactions on Information and System Security (TISSEC)*, *18*(1), pp.1-32.

8. Chowdhury, N., 2024. Analyzing password strength: A combinatorial entropy approach. In *Proceedings of ACM Conference (Conference 2017)*.