# Name : Mukesh Kumar

CIN: FIT/FEB25/CS583

# Password Analyzer Tool - Report

## Introduction

This report explains the functionality and effectiveness of the Password Analyzer Tool. The tool is designed using Python with Tkinter for GUI, regex for pattern matching, and hashlib for encryption. It evaluates password strength and provides a secure hash of the password.

## Algorithm Explanation

The tool follows these steps:

1. User Input: The user enters a password in the GUI.

2. Regex-based Analysis: The password is evaluated against multiple criteria:

- Length (Minimum 8 characters)

- Presence of at least one lowercase letter

- Presence of at least one uppercase letter

- Presence of at least one digit

- Presence of at least one special character

3. Strength Assessment : Based on failed criteria, the tool assigns a strength level from 'Very Weak' to 'Very Strong'.

4. Password Hashing : The password is hashed using the SHA-256 algorithm for security.

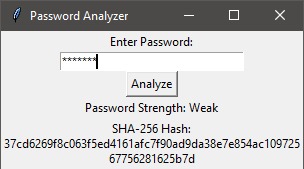
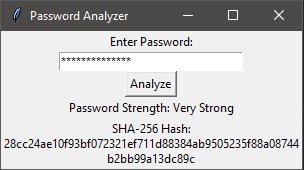
5. Results Display : The GUI displays the password strength and its hashed value.

## Effectiveness of the Algorithm

The password strength analysis uses regex-based pattern matching, ensuring accurate evaluation of password security. SHA-256 hashing provides a secure way to store passwords without exposing them in plaintext. By providing a user-friendly GUI, this tool allows users to assess their passwords easily and improve their security practices.

## Conclusion

This Password Analyzer Tool is an effective solution for users to test and strengthen their passwords. It leverages regex for analysis and hashlib for security, making it a reliable utility for personal or organizational use.



Import tkinter as tk

from tkinter import messagebox

import re

import hashlib

# Function to analyze password strength

def analyze\_password():

password = entry.get()

# Regex pattern checks

length\_ok = len(password) >= 8

lowercase\_ok = re.search(r'[a-z]', password) is not None

uppercase\_ok = re.search(r'[A-Z]', password) is not None

digit\_ok = re.search(r'\d', password) is not None

special\_char\_ok = re.search(r'[\W\_]', password) is not None

# Calculate strength level

passed\_criteria = sum([length\_ok, lowercase\_ok, uppercase\_ok, digit\_ok, special\_char\_ok])

strength\_levels = ["Very Weak", "Weak", "Moderate", "Strong", "Very Strong"]

strength = strength\_levels[passed\_criteria - 1] if passed\_criteria > 0 else "Very Weak"

# Hash the password

hashed\_password = hashlib.sha256(password.encode()).hexdigest()

# Display results

result\_label.config(text=f"Password Strength: {strength}")

hash\_label.config(text=f"SHA-256 Hash:\n{hashed\_password}")

# GUI Setup

root = tk.Tk()

root.title("Password Analyzer")

tk.Label(root, text="Enter Password:").pack()

entry = tk.Entry(root, show="\*", width=30)

entry.pack()

analyze\_btn = tk.Button(root, text="Analyze", command=analyze\_password)

analyze\_btn.pack()

result\_label = tk.Label(root, text="")

result\_label.pack()

hash\_label = tk.Label(root, text="", wraplength=300)

hash\_label.pack()

root.mainloop()