PCGM

PASSWORD CHECKER,
GENERATOR, AND MANAGER

TEAM ATHENA

Section: EQ3

Brent Calado

Ezekiel Santiago

James Tepace

INTRODUCTION

The program is a term project presented to Mr. Ramon Stephen L. Ruiz in partial fulfillment of the requirements for the course Programming Logic and Design (PROLOGI)

The program aims to test the strength of user-generated passwords, as well as create computer-generated passwords if the user-generated passwords are weak and store them through the program's password managing capabilities. Through this project, certain issues regarding network security may be resolved.

SPECIAL FEATURES

- Clearing of fields in the program GUI.
- Ability to copy password to clipboard.
- User-friendly and creative GUI with easy-to-use interface elements.
- Save files using user-inputted file names.

PSEUDOCODE

```
// Importing of modules
import random
import string
from tkinter import *
from tkinter import messagebox

Module main()
        Set self.master = master
        Set master.title("PCGM: Password Creator, Generator, and Manager")
        Set self.master.geometry("350x400")
        Set self.master.config(bg="pink")
```

// Create labels and entry widgets for website/application, username/email/phone number and password
Set self.label0 = Label(master, text="Website/Application:")
Display self.label0.grid(row=0, sticky=E)
Set self.entry0 = Entry(master)
Display self.entry0.grid(row=0, column=1)
Set self.label1 = Label(master, text="Username/Email/Phone number:")
Display self.label1.grid(row=1, sticky=W)
Set self.entry1 = Entry(master)
Display self.entry1.grid(row=1, column=1)

Set self.label2 = Label(master, text="Password:")
Display self.label2.grid(row=2, sticky=E)
Set self.entry2 = Entry(master, show="")
Display self.entry2.grid(row=2, column=1)

// Create button for checking password strength and generating a new password
Set self.button1 = Button(master, text="Check Password Strength", command=self.check_password_strength,bg="yellow", fg="black")
Display self.button1.grid(row=3, column=0, pady=10)
Call check_password_strength(self)

Set self.label4 = Label(master, text="Password Length:")
Display self.label4.grid(row=4, sticky=E)
Set self.pass_len = Entry(master)
Display self.pass_len.grid(row=4, column=1)

Set self.button2 = Button(master, text="Generate New Password", command=self.generate_password, bg="green", fg="white")
Display self.button2.grid(row=5, column=1, pady=10)
Call generate_password(self)

// Create label for displaying password strength and feedback
Set self.label3 = Label(master, text="")
Display self.label3.grid(row=3, column = 1, columnspan=2)

// Button to save login credentials and password to a text file
Set self.button3 = Button(master, text="Save Credentials", command=self.save_credentials, bg="maroon", fg="white")
Display self.button3.grid(row=7, column=0, pady=10)

Set self.label5 = Label(master, text="Enter File Name and extension:")
Display self.label5.grid(row=8, sticky=E)
Set self.save = Entry(master)
Display self.save.grid(row=8, column=1)
Call save_credentials(self)

// Create button for clearing the entry fields
Set self.clear_button = Button(master, text="Clear", command=self.clear_fields, bg='red', fg='white')
Display self.clear_button.grid(row=9, column=1, pady=10)
Call clear_fields(self)

// Create button for copying the password to the clipboard
Set self.copy_button = Button(master, text="Copy to Clipboard", command=self.copy_to_clipboard, bg="blue", fg="white")
Display self.copy_button.grid(row=9, column=0, pady=10)
Call copy_to_clipboard(self)
End Module

```
// Module that checks the strength of a password
Module check_password_strength(self)
    Set password = self.entry2.get()
    If len(password) < 8 Then
        Display self.label3.configure(text="Weak password: too short")
    Else If len(password) > 32 Then
        Display self.label3.configure(text="Weak password: too long")
    Else If not any(char.isdigit() for char in password) Then
        Display self.label3.configure(text="Weak password: no digits")
    Else If not any(char.isupper() for char in password) Then
        Display self.label3.configure(text="Weak password: no uppercase letters")
    Else If not any(char.islower() for char in password) Then
        Display self.label3.configure(text="Weak password: no lowercase letters")
    Else If not any(char in string.punctuation for char in password) Then
        Display self.label3.configure(text="Weak password: no special characters")
    Else
        Display self.label3.configure(text="Strong password!")
    End If
End Module
```

```
// Module that generates a new password
Module generate_password(self)
    Try
        Set upper = string.ascii_uppercase
        Set lower = string.ascii_lowercase
        Set numbers = string.digits
        Set symbols = string.punctuation
        Set all_chars = upper + lower + numbers + symbols
        Set password_length = int(self.pass_len.get())
        Set password = ''.join(random.sample(all_chars, password_length))
        Set self.entry2.delete(0, END)
        Set self.entry2.insert(0, password)
        Display self.label3.configure(text="Generated password: " + password)

        // Save the login credentials and password to a file
       Try
            Set name = self.save.get()
            With open(name, "a") as file:
            file.write( self.entry0.get() + ", " + self.entry1.get() + ", " + password + "\n")
            Display messagebox.showinfo("New Password Generated", f"New
            password has been generated and saved to {name}")
```

```
        // Raise error if no file name is inputted
        Except
            Display messagebox.showinfo("Error!", "Error! No file name inputted.")

    // Raise error if no password length is inputted
    Except
        Display messagebox.showinfo("Error!", "Error! No password length inputted.")
End Module
```

```
// Module that clears the fields
Module clear_fields(self)
        Set self.entry0.delete(0, END)
        Set self.entry1.delete(0, END)
        Set self.entry2.delete(0, END)
        Set self.pass_len.delete(0, END)
        Set self.save.delete(0, END)
        Set self.label3.configure(text="")
End Module
```

```
// Module that saves the credentials to a new file
Module save_credentials(self)
    Try
        Set file_name = self.save.get()
        With open(file_name, "a") as file:
        file.write( self.entry0.get() + ", " + self.entry1.get() + ", " + self.entry2.get() + "\n")
        Display messagebox.showinfo("Credentials saved", f"Username and password saved to {file_name}")

    // Raise error if no file name is inputted
    Except
        Display messagebox.showinfo("Error!", "Error! No file name inputted.")
End Module
```

// Module that copies the password to the clipboard of the user

```
Module copy_to_clipboard(self):
        Set password = self.entry2.get()
        Set self.master.clipboard_clear()
        Set self.master.clipboard_append(password)
        Set self.master.update()
        Display messagebox.showinfo("Copy to Clipboard", "Password has been copied to the clipboard")
End Module
```

DEMONSTRATION

THANK YOU!

ATHENA

brent_calado@dlsu.edu.ph

ezekiel_santiago@dlsu.edu.ph

james_tepace@dlsu.edu.ph