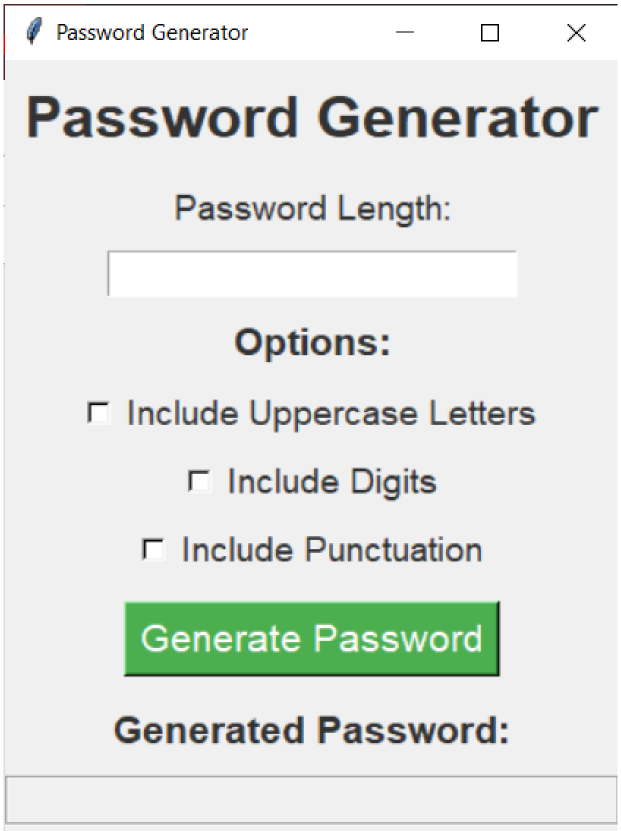
Task 2:- Password Generator

Code:-

```
import random
import tkinter as tk
from tkinter import messagebox

def generate_password(length, use_uppercase, use_digits, use_punctuation):
    characters = 'abcdefghijklmnopqrstuvwxyz'
    if use_uppercase:
        characters += 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
    if use_digits:
        characters += '0123456789'
    if use_punctuation:
        characters += '!@#$%^&*()-_=+[]{}|;:,.<>?/'

    password = ''.join(random.choice(characters) for _ in range(length))
    return password

class PasswordGeneratorApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Password Generator")

        # Set background color
        self.root.configure(bg="#f0f0f0")

        # Title label
        tk.Label(root, text="Password Generator", font=("Helvetica", 24, "bold"), bg="#f0f0f0",
fg="#333333").pack(pady=10)

        # Password length input
        tk.Label(root, text="Password Length:", font=("Helvetica", 14), bg="#f0f0f0",
fg="#333333").pack(pady=5)
        self.length_entry = tk.Entry(root, font=("Helvetica", 14))
        self.length_entry.pack(pady=5)

        # Options section
        tk.Label(root, text="Options:", font=("Helvetica", 16, "bold"), bg="#f0f0f0",
fg="#333333").pack(pady=5)

        self.uppercase_var = tk.BooleanVar()
        tk.Checkbutton(root, text="Include Uppercase Letters", variable=self.uppercase_var,
font=("Helvetica", 14), bg="#f0f0f0", fg="#333333").pack(pady=2)
```

```
        self.digits_var = tk.BooleanVar()
        tk.Checkbutton(root, text="Include Digits", variable=self.digits_var, font=("Helvetica", 14), bg="#f0f0f0", fg="#333333").pack(pady=2)

        self.punctuation_var = tk.BooleanVar()
        tk.Checkbutton(root, text="Include Punctuation", variable=self.punctuation_var, font=("Helvetica", 14), bg="#f0f0f0", fg="#333333").pack(pady=2)

        # Generate button
        tk.Button(root, text="Generate Password", command=self.generate_password, font=("Helvetica", 16), bg="#4CAF50", fg="white", activebackground="#45a049").pack(pady=10)

        # Display generated password
        tk.Label(root, text="Generated Password:", font=("Helvetica", 16, "bold"), bg="#f0f0f0", fg="#333333").pack(pady=5)
        self.password_display = tk.Entry(root, state='readonly', font=("Helvetica", 14), bd=2, relief="groove", width=30)
        self.password_display.pack(pady=5)

    def generate_password(self):
        try:
            length = int(self.length_entry.get())
            if length <= 0:
                raise ValueError("Password length must be a positive integer.")
        except ValueError as e:
            messagebox.showerror("Invalid Input", str(e))
            return

        use_uppercase = self.uppercase_var.get()
        use_digits = self.digits_var.get()
        use_punctuation = self.punctuation_var.get()

        password = generate_password(length, use_uppercase, use_digits, use_punctuation)
        self.password_display.config(state='normal')
        self.password_display.delete(0, tk.END)
        self.password_display.insert(0, password)
        self.password_display.config(state='readonly')

if __name__ == "__main__":
    root = tk.Tk()
    app = PasswordGeneratorApp(root)
    root.mainloop()
```

Output:-



Password Generator

Password Generator

Password Length:

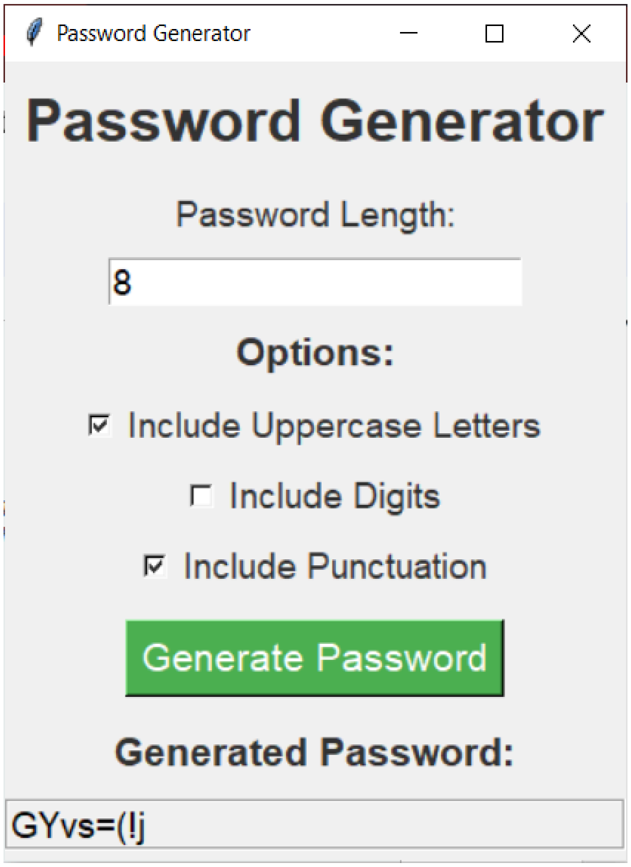
Options:

□ Include Uppercase Letters

□ Include Digits

□ Include Punctuation

Generated Password:



Password Generator

Password Generator

Password Length: 8

Options:

☑ Include Uppercase Letters

□ Include Digits

☑ Include Punctuation

Generated Password: GYvs=(!j