```
import random
import string
import sqlite3
import re
from tkinter import *
from tkinter import messagebox

# Ensure users.db exists and create table if not
with sqlite3.connect("users.db") as db:
    cursor = db.cursor()
    cursor.execute("CREATE TABLE IF NOT EXISTS users (Username TEXT NOT NULL, GeneratedPassword TEXT NOT
    db.commit()

# GUI Class
class GUI:
    def __init__(self, master):
        self.master = master
        self.master.title('Password Generator')
        self.master.geometry('660x500')
        self.master.config(bg='#FF8000')
        self.master.resizable(False, False)

        # Variables
        self.n_username = StringVar()
        self.n_passwordlen = IntVar()
        self.n_generatedpassword = StringVar()

        # Layout
        Label(text="PASSWORD GENERATOR", anchor=N, fg='darkblue', bg='#FF8000',
              font="arial 20 bold underline").grid(row=0, column=1, pady=10)

        Label(text="Enter User Name:", font='times 15 bold', bg="#FF8000", fg='darkblue').grid(row=1, col
        self.textfield = Entry(textvariable=self.n_username, font='times 15', bd=6, relief='ridge')
        self.textfield.grid(row=1, column=1, padx=10)
        self.textfield.focus_set()

        Label(text="Enter Password Length:", font='times 15 bold', bg='#FF8000', fg='darkblue').grid(row=
        self.length_textfield = Entry(textvariable=self.n_passwordlen, font='times 15', bd=6, relief='rid
        self.length_textfield.grid(row=2, column=1, padx=10)

        Label(text="Generated Password:", font='times 15 bold', bg='#FF8000', fg='darkblue').grid(row=3,
        self.generated_password_textfield = Entry(textvariable=self.n_generatedpassword, font='times 15',
                                                  relief='ridge')
        self.generated_password_textfield.grid(row=3, column=1, padx=10)

        # Buttons
        Button(text="GENERATE PASSWORD", bd=3, relief='solid', padx=1, pady=1,
               font="Verdana 15 bold", fg='black', command=self.generate_pass).grid(row=4, column=1, pady

        Button(text="ACCEPT", bd=3, relief='solid', padx=1, pady=1,
               font="Helvetica 15 bold italic", fg='#458800', command=self.accept_fields).grid(row=5, col

        Button(text="RESET", bd=3, relief='solid', padx=1, pady=1,
               font="Helvetica 15 bold italic", fg='#458800', command=self.reset_fields).grid(row=6, colu
```

```
def generate_pass(self):
    upper = list("ABCDEFGHIJKLMNOPQRSTUVWXYZ")
    lower = list("abcdefghijklmnopqrstuvwxyz")
    chars = list("@#%&()?!")
    numbers = list("1234567890")

    name = self.textfield.get()
    try:
        length = int(self.length_textfield.get())
    except ValueError:
        messagebox.showerror("Error", "Password length must be a number.")
        return

    if name == "":
        messagebox.showerror("Error", "Name cannot be empty")
        return
    if not name.isalpha():
        messagebox.showerror("Error", "Name must be alphabetic")
        self.textfield.delete(0, END)
        return
    if length < 6:
        messagebox.showerror("Error", "Password must be at least 6 characters long")
        return

    u = random.randint(1, length - 3)
    c = random.randint(1, length - u - 2)
    l = 1
    n = length - u - c - l

    password = random.sample(upper, u) + random.sample(lower, l) +                    random.sample(c
    random.shuffle(password)
    gen_passwd = ''.join(password)

    self.n_generatedpassword.set(gen_passwd)

def accept_fields(self):
    username = self.n_username.get()
    password = self.n_generatedpassword.get()

    if username == "" or password == "":
        messagebox.showerror("Error", "Username or Password field is empty")
        return

    with sqlite3.connect("users.db") as db:
        cursor = db.cursor()
        cursor.execute("SELECT * FROM users WHERE Username = ?", (username,))
        if cursor.fetchall():
            messagebox.showerror("Error", "This username already exists! Please use another one.")
        else:
            cursor.execute("INSERT INTO users (Username, GeneratedPassword) VALUES (?, ?)", (username
            db.commit()
            messagebox.showinfo("Success!", "Password saved successfully")
```

```
    def reset_fields(self):
        self.textfield.delete(0, END)
        self.length_textfield.delete(0, END)
        self.generated_password_textfield.delete(0, END)

# Run the application
if __name__ == "__main__":
    root = Tk()
    pass_gen = GUI(root)
    root.mainloop()
```