import tkinter as tk

from tkinter import ttk, messagebox, filedialog

# from PIL import Image, ImageTk

class AdharCardForm(tk.Tk):

    def \_\_init\_\_(self):

        super().\_\_init\_\_()

        self.title("Aadhar Card Application Form")

        self.geometry("800x700")

        self.minsize(700, 650)

        self.configure(padx=20, pady=20)

        # Fonts and styles

        self.style = ttk.Style(self)

        self.style.configure("TLabel", font=("Segoe UI", 12))

        self.style.configure("TEntry", font=("Segoe UI", 12))

        self.style.configure("TButton", font=("Segoe UI", 12))

        self.style.configure("TRadiobutton", font=("Segoe UI", 12))

        self.style.configure("TCombobox", font=("Segoe UI", 12))

        # Title label

        title\_label = ttk.Label(self, text="Aadhar Card Application Form", font=("Segoe UI", 20, "bold"))

        title\_label.pack(pady=(0, 20))

        # Main frame for form content

        form\_frame = ttk.Frame(self)

        form\_frame.pack(fill=tk.BOTH, expand=True)

        # Configure grid columns for even spacing

        form\_frame.columnconfigure(0, weight=1)

        form\_frame.columnconfigure(1, weight=1)

        # Left side fields

        # Full Name

        ttk.Label(form\_frame, text="Full Name \*").grid(row=0, column=0, sticky=tk.W, pady=8, padx=8)

        self.fullname\_entry = ttk.Entry(form\_frame)

        self.fullname\_entry.grid(row=1, column=0, sticky=tk.EW, padx=8)

        # Date of Birth

        ttk.Label(form\_frame, text="Date of Birth (DD/MM/YYYY) \*").grid(row=2, column=0, sticky=tk.W, pady=8, padx=8)

        self.dob\_entry = ttk.Entry(form\_frame)

        self.dob\_entry.grid(row=3, column=0, sticky=tk.EW, padx=8)

        # Gender Radio Buttons

        ttk.Label(form\_frame, text="Gender \*").grid(row=4, column=0, sticky=tk.W, pady=8, padx=8)

        self.gender\_var = tk.StringVar()

        gender\_frame = ttk.Frame(form\_frame)

        gender\_frame.grid(row=5, column=0, sticky=tk.W, padx=8)

        ttk.Radiobutton(gender\_frame, text="Male", variable=self.gender\_var, value="Male").pack(side=tk.LEFT, padx=8)

        ttk.Radiobutton(gender\_frame, text="Female", variable=self.gender\_var, value="Female").pack(side=tk.LEFT, padx=8)

        ttk.Radiobutton(gender\_frame, text="Other", variable=self.gender\_var, value="Other").pack(side=tk.LEFT, padx=8)

        # Mobile Number

        ttk.Label(form\_frame, text="Mobile Number \*").grid(row=6, column=0, sticky=tk.W, pady=8, padx=8)

        self.mobile\_entry = ttk.Entry(form\_frame)

        self.mobile\_entry.grid(row=7, column=0, sticky=tk.EW, padx=8)

        # Email Address

        ttk.Label(form\_frame, text="Email Address").grid(row=8, column=0, sticky=tk.W, pady=8, padx=8)

        self.email\_entry = ttk.Entry(form\_frame)

        self.email\_entry.grid(row=9, column=0, sticky=tk.EW, padx=8)

        # Right side fields

        # Address Line 1

        ttk.Label(form\_frame, text="Address Line 1 \*").grid(row=0, column=1, sticky=tk.W, pady=8, padx=8)

        self.addr1\_entry = ttk.Entry(form\_frame)

        self.addr1\_entry.grid(row=1, column=1, sticky=tk.EW, padx=8)

        # Address Line 2

        ttk.Label(form\_frame, text="Address Line 2").grid(row=2, column=1, sticky=tk.W, pady=8, padx=8)

        self.addr2\_entry = ttk.Entry(form\_frame)

        self.addr2\_entry.grid(row=3, column=1, sticky=tk.EW, padx=8)

        # City/Town

        ttk.Label(form\_frame, text="City/Town \*").grid(row=4, column=1, sticky=tk.W, pady=8, padx=8)

        self.city\_entry = ttk.Entry(form\_frame)

        self.city\_entry.grid(row=5, column=1, sticky=tk.EW, padx=8)

        # Pincode

        ttk.Label(form\_frame, text="Pincode \*").grid(row=6, column=1, sticky=tk.W, pady=8, padx=8)

        self.pincode\_entry = ttk.Entry(form\_frame)

        self.pincode\_entry.grid(row=7, column=1, sticky=tk.EW, padx=8)

        # Upload Photograph

        ttk.Label(form\_frame, text="Upload Photograph").grid(row=8, column=1, sticky=tk.W, pady=8, padx=8)

        photo\_frame = ttk.Frame(form\_frame)

        photo\_frame.grid(row=9, column=1, sticky=tk.W, padx=8)

        self.photo\_path\_var = tk.StringVar()

        self.photo\_label = ttk.Label(photo\_frame, text="No file selected", width=30)

        self.photo\_label.pack(side=tk.LEFT)

        ttk.Button(photo\_frame, text="Browse", command=self.browse\_photo).pack(side=tk.LEFT, padx=10)

        # Submitted data display area

        self.result\_text = tk.Text(self, height=8, font=("Segoe UI", 11))

        self.result\_text.pack(fill=tk.BOTH, expand=False, pady=(20, 0))

        # Buttons frame

        button\_frame = ttk.Frame(self)

        button\_frame.pack(pady=20)

        ttk.Button(button\_frame, text="Submit", command=self.submit\_form).pack(side=tk.LEFT, padx=15)

        ttk.Button(button\_frame, text="Reset", command=self.reset\_form).pack(side=tk.LEFT, padx=15)

    def browse\_photo(self):

        filetypes = [("Image files", "\*.jpg \*.jpeg \*.png")]

        filepath = filedialog.askopenfilename(title="Select Photograph", filetypes=filetypes)

        if filepath:

            self.photo\_path\_var.set(filepath)

            display\_text = filepath.split("/")[-1]

            self.photo\_label.config(text=display\_text)

    def submit\_form(self):

        full\_name = self.fullname\_entry.get().strip()

        dob = self.dob\_entry.get().strip()

        gender = self.gender\_var.get()

        mobile = self.mobile\_entry.get().strip()

        email = self.email\_entry.get().strip()

        addr1 = self.addr1\_entry.get().strip()

        addr2 = self.addr2\_entry.get().strip()

        city = self.city\_entry.get().strip()

        pincode = self.pincode\_entry.get().strip()

        photo = self.photo\_path\_var.get()

        # Basic validation

        if not full\_name:

            messagebox.showerror("Error", "Full Name is required.")

            return

        if not dob:

            messagebox.showerror("Error", "Date of Birth is required.")

            return

        if not gender:

            messagebox.showerror("Error", "Gender is required.")

            return

        if not mobile or not mobile.isdigit() or len(mobile) != 10:

            messagebox.showerror("Error", "Valid 10-digit Mobile Number is required.")

            return

        if not addr1:

            messagebox.showerror("Error", "Address Line 1 is required.")

            return

        if not city:

            messagebox.showerror("Error", "City/Town is required.")

            return

        if not pincode or not pincode.isdigit() or len(pincode) != 6:

            messagebox.showerror("Error", "Valid 6-digit Pincode is required.")

            return

        # Clear previous result text

        self.result\_text.delete(1.0, tk.END)

        # Format output

        result = (

            f"Full Name: {full\_name}\n"

            f"Date of Birth: {dob}\n"

            f"Gender: {gender}\n"

            f"Mobile Number: {mobile}\n"

            f"Email Address: {email if email else 'N/A'}\n"

            f"Address Line 1: {addr1}\n"

            f"Address Line 2: {addr2 if addr2 else 'N/A'}\n"

            f"City/Town: {city}\n"

            f"Pincode: {pincode}\n"

            f"Photograph: {photo if photo else 'Not uploaded'}\n"

        )

        self.result\_text.insert(tk.END, result)

        messagebox.showinfo("Success", "Form submitted successfully!")

    def reset\_form(self):

        self.fullname\_entry.delete(0, tk.END)

        self.dob\_entry.delete(0, tk.END)

        self.gender\_var.set("")

        self.mobile\_entry.delete(0, tk.END)

        self.email\_entry.delete(0, tk.END)

        self.addr1\_entry.delete(0, tk.END)

        self.addr2\_entry.delete(0, tk.END)

        self.city\_entry.delete(0, tk.END)

        self.pincode\_entry.delete(0, tk.END)

        self.photo\_path\_var.set("")

        self.photo\_label.config(text="No file selected")

        self.result\_text.delete(1.0, tk.END)

if \_\_name\_\_ == "\_\_main\_\_":

    app = AdharCardForm()

    app.mainloop()