In [ ]:

```
#! python3
import sys
import random
import datetime
import tkinter as tk
from tkinter import PhotoImage
from tkinter import messagebox

import requests
from PIL import ImageTk, Image

api_key = 'd0f4215f39312e5de368ee8edad554b8'

class CustomFrame(tk.Frame):
    def __init__(self, parent, **kwargs):
        tk.Frame.__init__(self, parent, **kwargs)

        width = kwargs.get('width', 90)
        height = kwargs.get('height', 130)
        self.configure(width=width, height=height, padx=10)

class CustomLabel(tk.Label):
    def __init__(self, parent, **kwargs):
        tk.Label.__init__(self, parent, **kwargs)

        width = kwargs.get('width', 90)
        height = kwargs.get('height', 130)
        bg = kwargs.get('bg', 'Gray92')
        fg = kwargs.get('fg', 'black')
        self.configure(width=width, height=height, bg=bg, compound=tk.TOP,
                       font=('Arial', 13), wraplength=100)

class Application(tk.Frame):
    def __init__(self, master=None):
        super().__init__(master=master)
        self.master = master
        self.pack()

        self.label = tk.Label(self, image=bg_image)
        self.label.grid(row=0, column=0)
        self.label.grid_propagate(False)

        self.date = self.current_date()
        self.label_bg = 'Gray92'
        self.city = tk.StringVar()
        self.city.set('')

        self.draw_frames()
        self.draw_widgets()
        self.current_time()

        self.master.bind('<Return>', self.get_weather)

    def draw_frames(self):
        self.top = CustomFrame(self.label, width=490, height=50, bg='#355E3B')
        self.top.grid(row=0, column=0, columnspan=3)
        self.top.grid_propagate(False)

        self.search_frame = CustomFrame(self.label, width=490, height=40, bg='#355E3B')
        self.search_frame.grid(row=1, column=0, columnspan=3, pady=(2,0))
        self.search_frame.grid_propagate(False)

        self.datetime_frame = CustomFrame(self.label, width=145, height=95, bg='#355E3B')
        self.datetime_frame.grid(row=0, column=3, rowspan=2, padx=3)
        self.datetime_frame.grid_propagate(False)

        # windspeed frame
        self.windspeed = CustomFrame(self.label, bg=self.label_bg)
        self.windspeed.grid(row=2, column=3, pady=(100, 20))
        self.windspeed_lbl = CustomLabel(self.windspeed, image=wind_icon, text='\nWindspeed')
        self.windspeed_lbl.grid(row=0, column=0, padx=(5,2), sticky='S')

        # weather frame
        self.weather = CustomFrame(self.label, bg=self.label_bg)
        self.weather.grid(row=3, column=0)
        self.weather_lbl = CustomLabel(self.weather, image=clear_weather_icon, text='\nWeather')
        self.weather_lbl.grid(row=0, column=0)

        # temperature frame
        self.temperature = CustomFrame(self.label, bg=self.label_bg)
        self.temperature.grid(row=3, column=1)
        self.temperature_lbl = CustomLabel(self.temperature, image=high_temp_icon, text='\nTemperature')
        self.temperature_lbl.grid(row=0, column=0, padx=(5,2))

        # humidity frame
        self.humidity = CustomFrame(self.label, bg=self.label_bg)
        self.humidity.grid(row=3, column=2)
        self.humidity_lbl = CustomLabel(self.humidity, image=humidity_icon, text='\nHumidity' )
        self.humidity_lbl.grid(row=0, column=0, padx=(5,10))

        # pressure frame
        self.pressure = CustomFrame(self.label, bg=self.label_bg)
        self.pressure.grid(row=3, column=3)
        self.pressure_lbl = CustomLabel(self.pressure, text='\nPressure', image=pressure_icon)
        self.pressure_lbl.grid(row=0, column=0, padx=(5,2))

    def draw_widgets(self):
        # self.top
        self.app_label = tk.Label(self.top, text='Current Weather', font=('Arial', 20)
                            , bg='#355E3B', fg='white')
        self.app_label.grid(row=0, column=0, ipady=10, ipadx=10)

        # self.datetime_frame
        self.date_label = tk.Label(self.datetime_frame, text=self.date , font=('Arial', 15, 'bold')
                            , bg='#355E3B', fg='white', anchor='w')
        self.date_label.grid(row=0, column=0, ipady=14, ipadx=0)

        self.time_label = tk.Label(self.datetime_frame , font=('Calibri', 14)
                            , bg='#355E3B', fg='white', anchor='w')
        self.time_label.grid(row=1, column=0, ipady=0, ipadx=0)

        # self.search_frame
        self.search_label = tk.Label(self.search_frame, text='Search City : '
                            ,bg='#355E3B', fg='white', anchor='w',
                             font=('Arial', 11))
        self.search_label.grid(row=0, column=0, ipady=8, padx=(10,2))

        self.entry = tk.Entry(self.search_frame, bg='#355E3B', relief=tk.FLAT,
                                borderwidth=1, textvariable=self.city, fg='white')
        self.entry.focus_set()
        self.entry.grid(row=0, column=1, ipady=2)

        self.search = tk.Button(self.search_frame, image=search_icon,
                        command=self.weather_search, relief=tk.FLAT, bg='DodgerBlue2')
        self.search.grid(row=0, column=2, padx=5, ipady=1)

        self.city_label = tk.Label(self.search_frame, text=''
                            ,bg='#355E3B', fg='white', anchor='c',
                             font=('Arial', 10, 'bold'), width=19)
        self.city_label.grid(row=0, column=3, ipady=3,  padx=(41,0))

    def current_time(self):
        dt = datetime.datetime.now()
        self.time_label['text'] = dt.strftime('%I:%M:%S %p')
        self.time_label.after(1000, self.current_time)

    def current_date(self):
        dt = datetime.datetime.today()
        return dt.strftime('%d %b, %Y')

    def weather_search(self):
        self.get_weather()

    def get_weather(self, event=None):
        city = self.city.get()
        if len(city) > 2:
            url = f'https://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}'
            try:
                self.update()
                r = requests.get(url)
                data = r.json()

                weather = data['weather'][0]['description']
                weather = weather.lower()
                temp = round(data['main']['temp'] - 273.15, 2)
                if len(weather.split()) == 1:
                    weather = '\n' + weather
                else:
                    weather = '\n'.join(weather.split())
                self.weather_lbl['text'] = f"{weather}"
                self.temperature_lbl['text'] = f"\n{temp} C"
                self.windspeed_lbl['text'] = f"\n{data['wind']['speed']} m/s"
                self.humidity_lbl['text'] = f"\n{data['main']['humidity']} %"
                self.pressure_lbl['text'] = f"\n{data['main']['pressure']} hPa"

                if temp <= 18:
                    self.temperature_lbl['image'] = low_temp_icon
                else:
                    self.temperature_lbl['image'] = high_temp_icon

                if 'thunder' in weather:
                    self.weather_lbl['image'] = thunderstorm_icon
                elif 'cloud' in weather:
                    self.weather_lbl['image'] = cloudy_icon
                elif 'snow' in weather:
                    self.weather_lbl['image'] = snow_icon
                elif 'drizzle' in weather or 'rain' in weather:
                    self.weather_lbl['image'] = drizzle_icon
                elif ('mist' in weather or 'haze' in weather or 'fog' in weather
                        or 'smoke' in weather):
                    self.weather_lbl['image'] = mist_icon
                elif 'hail' in weather:
                    self.weather_lbl['image'] = hail_icon
                else:
                    self.weather_lbl['image'] = clear_weather_icon

                self.city_label['text'] = f'Weather in {city.capitalize()}'

            except KeyError:
                messagebox.showerror('Weatherzilla', 'No such city in database')
            except:
                messagebox.showerror('Weatherzilla', 'No internet Connection')

            self.entry.delete(0, tk.END)

if __name__ == '__main__':
    root = tk.Tk()
    root.title('Weather Aryan')

    if not api_key:
        root.withdraw()
        messagebox.showerror('WeatherAryan', 'OpenWeatherMap Api Key is required\n to use this App')
        sys.exit(0)

    bg_list = [f'wallpapers/bg{i}.png' for i in range(1,7)]
    bg = Image.open(random.choice(bg_list))
    bg_image = ImageTk.PhotoImage(bg)

    search_icon = PhotoImage(file='icons/search.png')

    clear_weather_icon = PhotoImage(file='icons/clear.png')
    clouds = Image.open('icons/clouds.png')
    cloudy_icon = ImageTk.PhotoImage(clouds)
    high_temp_icon = PhotoImage(file='icons/high_temp.png')
    low_temp_icon = PhotoImage(file='icons/low_temp.png')
    humidity_icon = PhotoImage(file='icons/humidity.png')
    pressure_icon = PhotoImage(file='icons/pressure.png')
    wind_icon = PhotoImage(file='icons/wind.png')
    thunderstorm_icon = PhotoImage(file='icons/thunderstorm.png')
    snow_icon = PhotoImage(file='icons/snow.png')
    drizzle_icon = PhotoImage(file='icons/drizzle.png')
    mist_icon = PhotoImage(file='icons/mist.png')
    hail_icon = PhotoImage(file='icons/hail.png')

    app = Application(master=root)
    app.mainloop()
```

In [ ]: