Let’s go through the code step by step to understand how the weather app works.

**1. Importing Required Modules**

python

Copy code

import requests

import tkinter as tk

from tkinter import messagebox

from PIL import Image, ImageTk

import ttkbootstrap

* **requests**: A library for sending HTTP requests to interact with web APIs (like OpenWeatherMap).
* **tkinter**: Python's standard GUI (Graphical User Interface) package for building desktop applications.
* **messagebox**: A submodule of tkinter that provides simple pop-up dialogues (e.g., error messages).
* **PIL (Python Imaging Library)**: Specifically, Image and ImageTk are used to handle and display images in the GUI.
* **ttkbootstrap**: A library that enhances the appearance of tkinter with modern themes and styles.

**2. Function to Get Weather Information**

python

Copy code

def get\_weather(city):

API\_Key = "e06c1aa420d107549b8a1ac567de2b0e"

url = f"https://api.openweathermap.org/data/2.5/weather?q={city}&appid={API\_Key}"

res = requests.get(url)

* **API\_Key**: The unique key provided by OpenWeatherMap to authenticate API requests.
* **url**: Constructs the API endpoint URL by embedding the city name and API key into the request URL.
* **requests.get(url)**: Sends an HTTP GET request to the API to fetch weather data for the specified city.

python

Copy code

if res.status\_code == 404:

messagebox.showerror("Error", "City not found")

return None

* **status\_code == 404**: Checks if the server returned a 404 error, meaning the city was not found. If so, it shows an error message and returns None.

python

Copy code

weather = res.json()

icon\_id = weather['weather'][0]['icon']

temperature = weather['main']['temp'] - 273.15

description = weather['weather'][0]['description']

city = weather['name']

country = weather['sys']['country']

* **res.json()**: Parses the JSON response from the API into a Python dictionary.
* **Extracting Data**: Retrieves the weather icon ID, temperature (converted from Kelvin to Celsius), description, city name, and country from the JSON response.

python

Copy code

icon\_url = f"https://openweathermap.org/img/wn/{icon\_id}@2x.png"

return (icon\_url, temperature, description, city, country)

* **icon\_url**: Constructs the URL for the weather icon.
* **return**: Returns a tuple containing all the relevant weather information.

**3. Function to Search for Weather**

python

Copy code

def search():

city = city\_entry.get()

result = get\_weather(city)

if result is None:

return

* **city\_entry.get()**: Gets the city name input from the user.
* **get\_weather(city)**: Calls the get\_weather function to fetch the weather data.
* **if result is None**: If the city wasn't found, the function exits early.

python

Copy code

icon\_url, temperature, description, city, country = result

location\_label.configure(text=f"{city},{country}")

* **Unpacking the Result**: The returned tuple from get\_weather is unpacked into individual variables.
* **location\_label.configure()**: Updates the label to display the city and country name.

python

Copy code

image = Image.open(requests.get(icon\_url, stream=True).raw)

icon = ImageTk.PhotoImage(image)

icon\_label.configure(image=icon)

icon\_label.image = icon

* **Loading the Weather Icon**: The icon is fetched from the icon\_url, converted into a format tkinter can display, and then applied to the icon\_label.
* **icon\_label.image = icon**: Keeps a reference to the image object to prevent it from being garbage collected.

python

Copy code

temperature\_label.configure(text=f"Temperature: {temperature:.2f}°C")

description\_label.configure(text=f"Description: {description}")

* **Update Labels**: The temperature and weather description labels are updated with the fetched data.

**4. Function to Show Info about PM Accelerator**

python

Copy code

def show\_info():

info\_text = (

"PM Accelerator is a professional service company that provides specialized "

"project management training and consulting services. We help organizations "

"achieve their business goals by delivering tailored project management solutions.\n\n"

"The Product Manager Accelerator Program is designed to support PM professionals "

"through every stage of their career. From students looking for entry-level jobs to "

"Directors looking to take on a leadership role, our program has helped over hundreds "

"of students fulfill their career aspirations.\n\n"

"Our Product Manager Accelerator community is ambitious and committed. Through our program, "

"they have learned, honed, and developed new PM and leadership skills, giving them a strong foundation "

"for their future endeavors."

)

messagebox.showinfo("About PM Accelerator", info\_text)

* **show\_info**: A function that, when called, displays a pop-up window with information about the PM Accelerator program using messagebox.showinfo().

**5. Creating the Main Window**

python

Copy code

root = ttkbootstrap.Window(themename="morph")

root.title("Weather App")

root.geometry("400x400")

* **ttkbootstrap.Window**: Creates the main application window with the "morph" theme.
* **root.title**: Sets the title of the window.
* **root.geometry**: Specifies the size of the window (400x400 pixels).

**6. Adding GUI Widgets**

python

Copy code

city\_entry = ttkbootstrap.Entry(root, font="Helvetica, 18")

city\_entry.pack(pady=10)

* **Entry**: A text entry widget for the user to input the city name.
* **pack(pady=10)**: Positions the widget in the window with some vertical padding.

python

Copy code

search\_button = ttkbootstrap.Button(root, text="Search", command=search, bootstyle="warning")

search\_button.pack(pady=10)

* **Button**: A button widget that, when clicked, calls the search() function to get weather data for the entered city.

python

Copy code

location\_label = tk.Label(root, font="Helvetica, 25")

location\_label.pack(pady=20)

* **Label**: Displays the city and country name after fetching the weather data.

python

Copy code

icon\_label = tk.Label(root)

icon\_label.pack()

* **Label**: Displays the weather icon.

python

Copy code

temperature\_label = tk.Label(root, font="Helvetica, 20")

temperature\_label.pack()

* **Label**: Displays the temperature.

python

Copy code

description\_label = tk.Label(root, font="Helvetica, 20")

description\_label.pack()

* **Label**: Displays the weather description.

python

Copy code

name\_label = tk.Label(root, text="Developed by Priya Harshe", font="Helvetica, 15")

name\_label.pack(pady=20)

* **Label**: Displays the developer's name.

python

Copy code

info\_button = ttkbootstrap.Button(root, text="Info", command=show\_info, bootstyle="info")

info\_button.pack(pady=10)

* **Button**: A button that displays information about the PM Accelerator when clicked.

**7. Start the GUI Main Loop**

python

Copy code

root.mainloop()

* **root.mainloop()**: Starts the Tkinter event loop, which keeps the window open and responsive until the user closes it.

**Summary**

This code creates a weather app using Python's tkinter library for the GUI. It fetches weather data from the OpenWeatherMap API, displays it in a user-friendly interface, and includes a button to display information about the PM Accelerator program. The app features modern styling with ttkbootstrap and handles various weather data like temperature, description, and icons.