JAIN DEEMED-TO-BE UNIVERSITY | SCHOOL OF COMPUTER SCIENCE AND IT

PROJECT: THE WEATHER CHANNEL

By

SNEHAL GHORPADE (22BCAR0373)

MOHAMMAD FAIZ AZIM KHAN (22BCAR0314)

RISHITA ROY (22BCAR0223)

KASHISH BHANDARI (22BCAR0211)

ESHA BHANDARI (22BCAR0206)

Guide:

PROF.KARTHIKEYAN

2023-2024

TOPIC: WEATHER FORECAST APP

INTRODUCTION

Introduction to the Python Tkinter project: The Weather Channel

Welcome to "The Weather Channel", your go-to companion for quick and essential weather updates. In a world that's constantly on the move, staying informed about the weather shouldn't be a hassle. That's why we've crafted a minimalistic weather application that delivers the crucial details you need at a glance.

Weather Snap focuses on four key elements – temperature, wind speed, atmospheric pressure, and humidity – providing you with a concise snapshot of the current weather conditions. We understand that your time is valuable, so we've designed an interface that is both intuitive and efficient.

Whether you're planning your day, preparing for an outdoor activity, or just curious about the weather around you, Weather Snap has you covered. Say goodbye to overwhelming charts and unnecessary clutter; our application delivers the essentials cleanly and straightforwardly.

Stay ahead of the weather curve with Weather Snap – where simplicity meets precision.

DEVELOPMENT TOOLS

- Coding language: Python
- Development Environment: IDLE(Python 3.12 64 bit),
- API keys: openweathermap.org,
- Debugging purposes: Chat GPT
- Images: Google

ADVANTAGES

The advantages of the Weather Channel:

- **Daily Planning and Convenience**: These apps provide detailed weather forecasts, helping individuals plan their day-to-day activities. Knowing whether it will rain, snow, or be sunny can affect clothing choices, transportation methods, and outdoor plans.
- **Travel and Vacation Planning**: Weather apps are crucial for travelers. They help in choosing the best times to visit certain destinations and in packing appropriately. Unexpected weather conditions can significantly affect travel plans, and being informed helps mitigate such risks.
- **Agriculture and Farming**: Farmers rely heavily on weather forecasts to make crucial decisions about planting, harvesting, and protecting crops from adverse weather conditions. Accurate forecasts can lead to better crop yields and prevent losses due to unforeseen weather events.
- **Safety and Emergency Preparedness**: Weather apps provide early warnings for severe weather conditions such as hurricanes, tornadoes, heavy snowfall, or flooding. This allows individuals and communities to take necessary precautions, secure property, and evacuate if needed, potentially saving lives.
- **Business Operations and Logistics**: Many businesses depend on weather forecasts for operational efficiency, especially those in sectors like construction, transportation, and shipping. Knowing the weather in advance helps in scheduling work, preventing damage to goods, and minimizing delays.
- **Outdoor Sports and Events**: Organizers of outdoor sports and events rely on weather forecasts to schedule events, ensure participant safety, and decide on

cancellations or postponements. Participants can also use this information to prepare appropriately for the conditions.

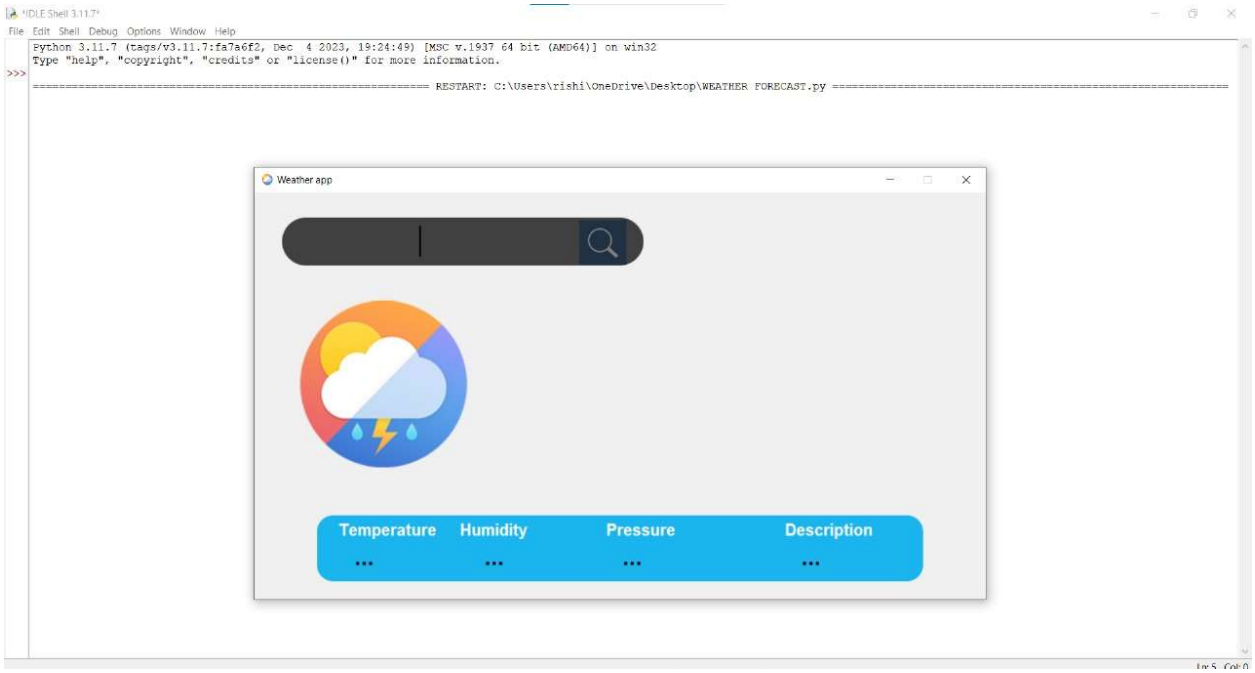
- **Energy Management**: Weather apps can aid in energy management by predicting periods of high or low demand based on temperature forecasts. This is particularly relevant for electric and heating utilities, which can adjust production accordingly.
- **Educational Tool**: Weather apps can serve as educational tools, helping users understand weather patterns, climate change, and the importance of meteorology in daily life. They can foster a greater appreciation for the environment and the complexities of the Earth's climate system.

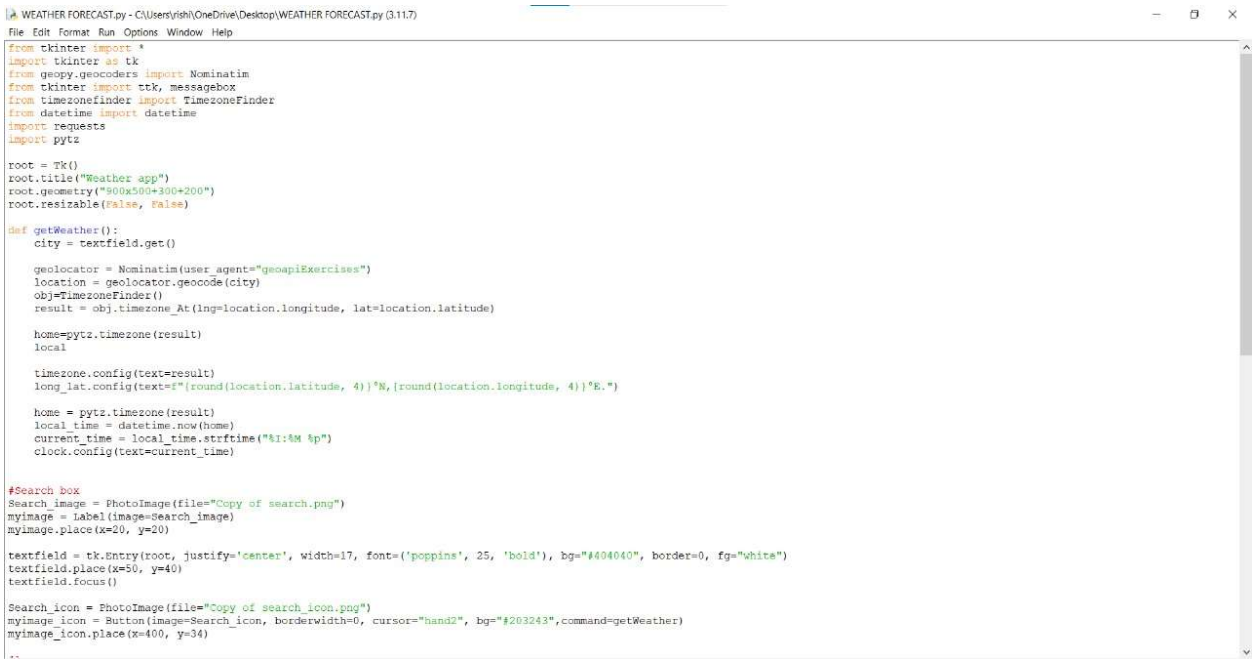
OBJECTIVES OF THE PROJECT

- **Provide Accurate and Up-to-Date Weather Information**: Offer real-time weather updates and forecasts to users, ensuring the highest possible accuracy using advanced meteorological data and models.
- **Enhance User Safety**: Issue timely warnings and alerts for extreme weather conditions such as storms, hurricanes, floods, and heatwaves to help users take necessary precautions to safeguard themselves and their properties.
- **Improve User Convenience and Decision Making**: Assist users in planning their daily activities, travel, and events by providing detailed weather forecasts, including temperature, wind speed, and humidity.
- **Offer Personalization**: Allow users to customize the app based on their location, preferences, and specific needs (e.g., alerts for certain weather conditions, tracking multiple locations) to enhance the user experience.
- **Support the Needs of Specific User Groups**: Cater to the specialized needs of various user groups, such as farmers, fishermen, outdoor enthusiasts, and event organizers, by providing targeted weather information relevant to their activities.
- **Promote Environmental Awareness**: Use the platform to encourage responsible behavior toward the environment by highlighting the impact of weather and climate patterns on the planet.

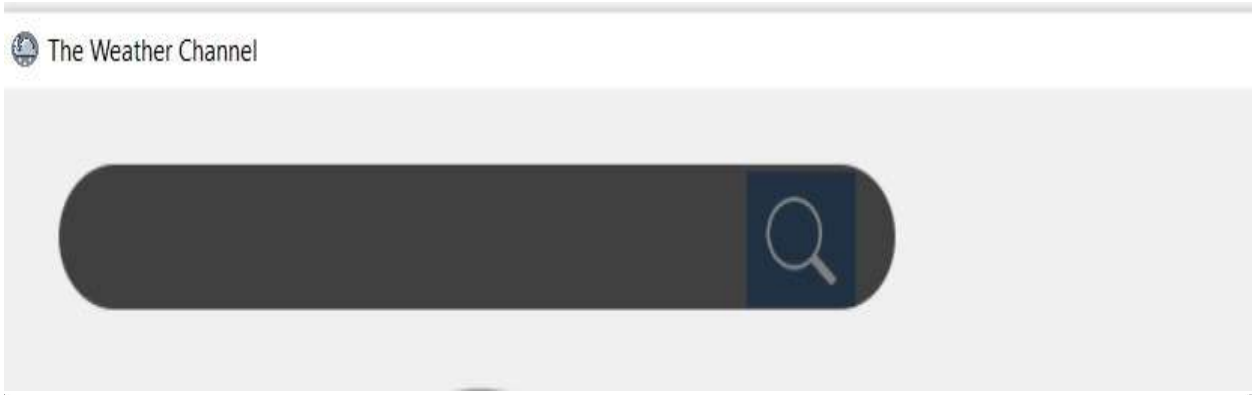
SCREENSHOTS

1. Building of the Application





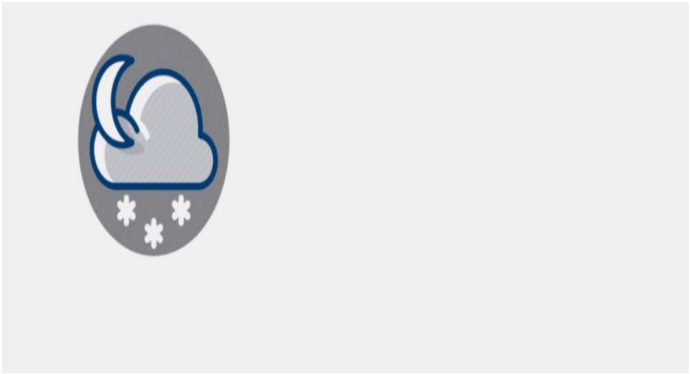
2. Search box to search the weather forecast of any place.




```
#Search box
Search_image = PhotoImage(file="Copy of search.png")
myimage = Label(image=Search_image)
myimage.place(x=20, y=20)

textfield = tk.Entry(root, justify='center', width=17, font=('poppins', 25, 'bold'), bg="#404040", border=0, fg="white")
textfield.place(x=50, y=40)
textfield.focus()

Search_icon = PhotoImage(file="Copy of search_icon.png")
myimage_icon = Button(image=Search_icon, borderwidth=0, cursor="hand2", bg="#203243",command=getWeather)
myimage_icon.place(x=400, y=34)
```

3. Logo

```
#logo
image_icon = PhotoImage(file="forecast image.png")
root.iconphoto(False, image_icon)

Round_box = PhotoImage(file="forecast image.png")
Label(root, image=Round_box).place(x=156, y=110)
```



4. Box displaying the details about the weather forecast

```
#Bottom Box
Frame_image=PhotoImage(file="Copy of box.png")
frame_myimage=Label(image=Frame_image)
frame_myimage.pack(padx=5,pady=5,side=BOTTOM)
```

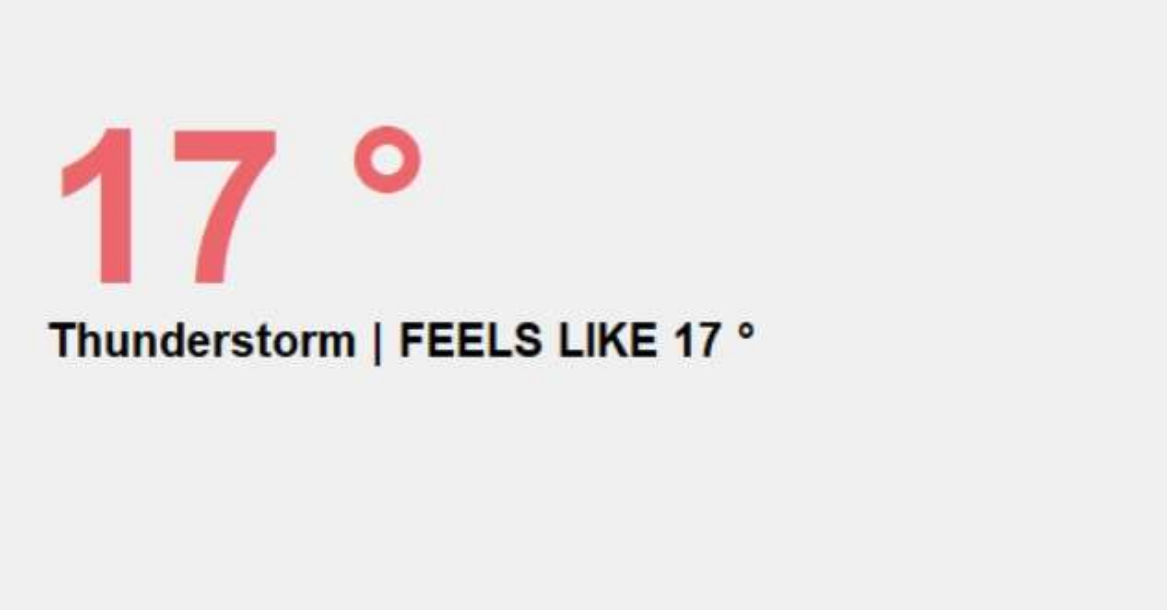
5. Displaying the current time of the place

```
#time
name=Label(root,font=("arial",15,"bold"))
name.place(x=30,y=100)
clock=Label(root,font=("Helvetica",20))
clock.place(x=50,y=175)

        home = pytz.timezone(result)
        local_time = datetime.now(home)
        current_time = local_time.strftime("%I:%M %p")
        clock.config(text=current_time)
        name.config(text="CURRENT WEATHER")
```

```
# label
label1 = Label(root, text="Wind", font=("Helvetica",15,'bold'),fg="white",bg="#1ab5ef")
label1.place(x=100, y=400)

label2 = Label(root, text="Humidity", font=("Helvetica", 15,'bold'), fg="white",bg="#1ab5ef")
label2.place(x=250, y=400)

label3 = Label(root, text="Pressure", font=("Helvetica", 15,'bold'), fg="white",bg="#1ab5ef")
label3.place(x=430, y=400)

label4 = Label(root, text="Description", font=("Helvetica", 15,'bold'), fg="white",bg="#1ab5ef")
label4.place(x=650, y=400)

t=Label(font=("arial",70,"bold"),fg="#ee666d")
t.place(x=400,y=150)
c=Label(font=("arial",15,'bold'))
c.place(x=400,y=250)

W=Label(text="...",font=("arial",20,"bold"),bg="#1ab5ef")
W.place(x=120,y=430)
H=Label(text="...",font=("arial",20,"bold"),bg="#1ab5ef")
H.place(x=280,y=430)
P=Label(text="...",font=("arial",20,"bold"),bg="#1ab5ef")
P.place(x=450,y=430)
D=Label(text="...",font=("arial",20,"bold"),bg="#1ab5ef")
D.place(x=670,y=430)
```

6. Temperature

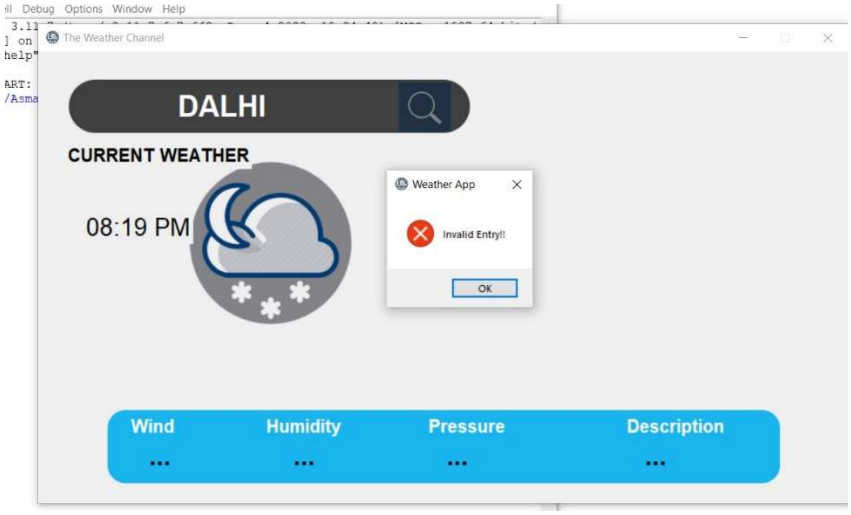
```
#weather
api="https://api.openweathermap.org/data/2.5/weather?q="+city+"&appid=48b43465bfec45bdf9bcde06064458e0"

json_data = requests.get(api).json()
condition = json_data['weather'][0]['main']
description = json_data['weather'][0]['description']
temp =int(json_data['main']['temp']-273.15)
pressure = json_data['main']['pressure']
humidity = json_data['main']['humidity']
wind = json_data['wind']['speed']

t.config(text=(temp,"°"))
c.config(text=(condition,"|","FEELS","LIKE",temp,"°"))

W.config(text=wind)
H.config(text=humidity)
D.config(text=description)
P.config(text=pressure)
```



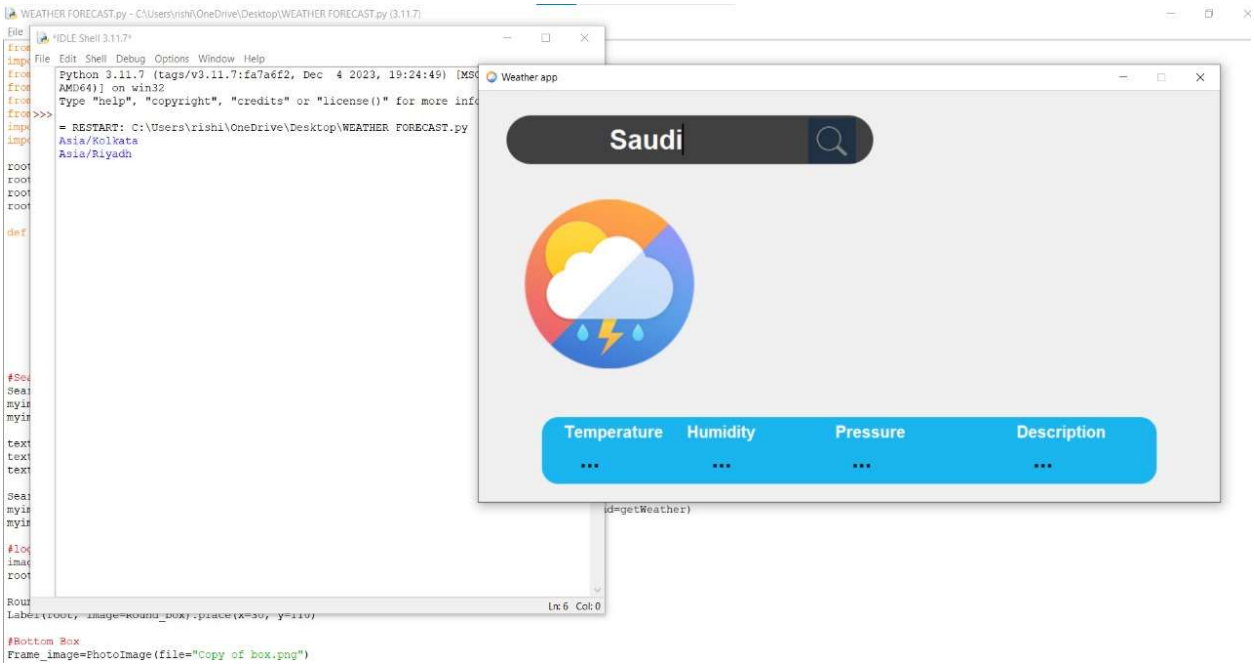
7. API key

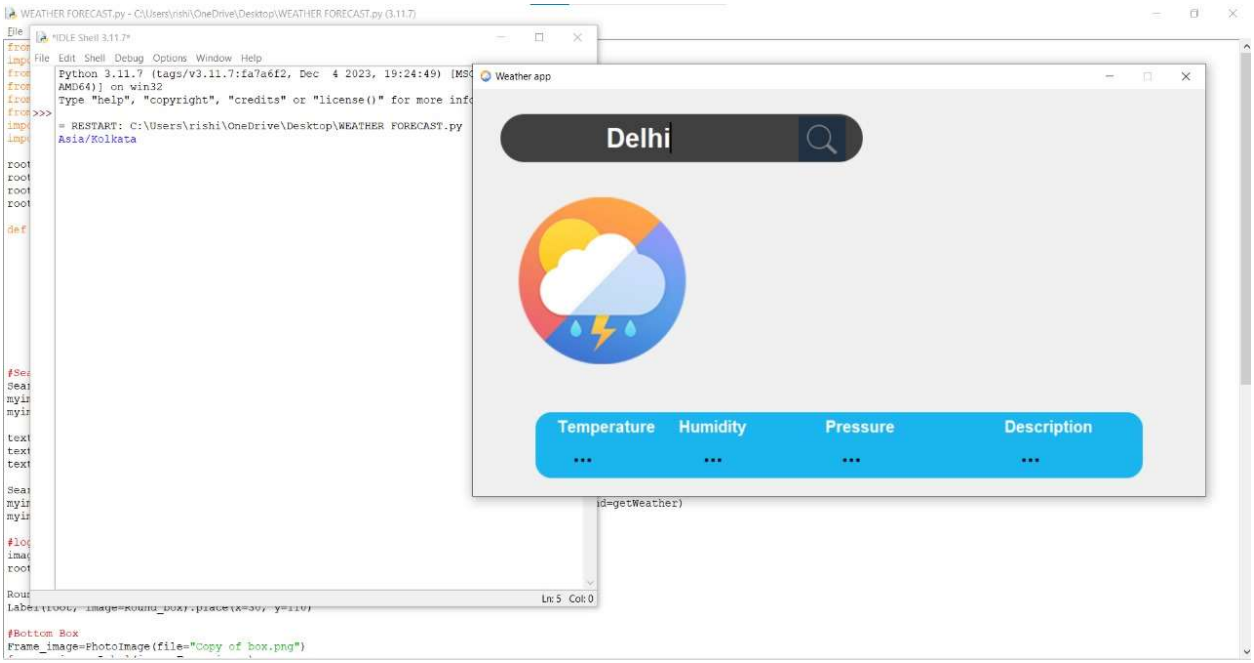
8. Wrong inputs

```
except Exception as e:
    messagebox.showerror("Weather App","Invalid Entry!!")
```



9. Detection of locations and regions





SOURCE CODE

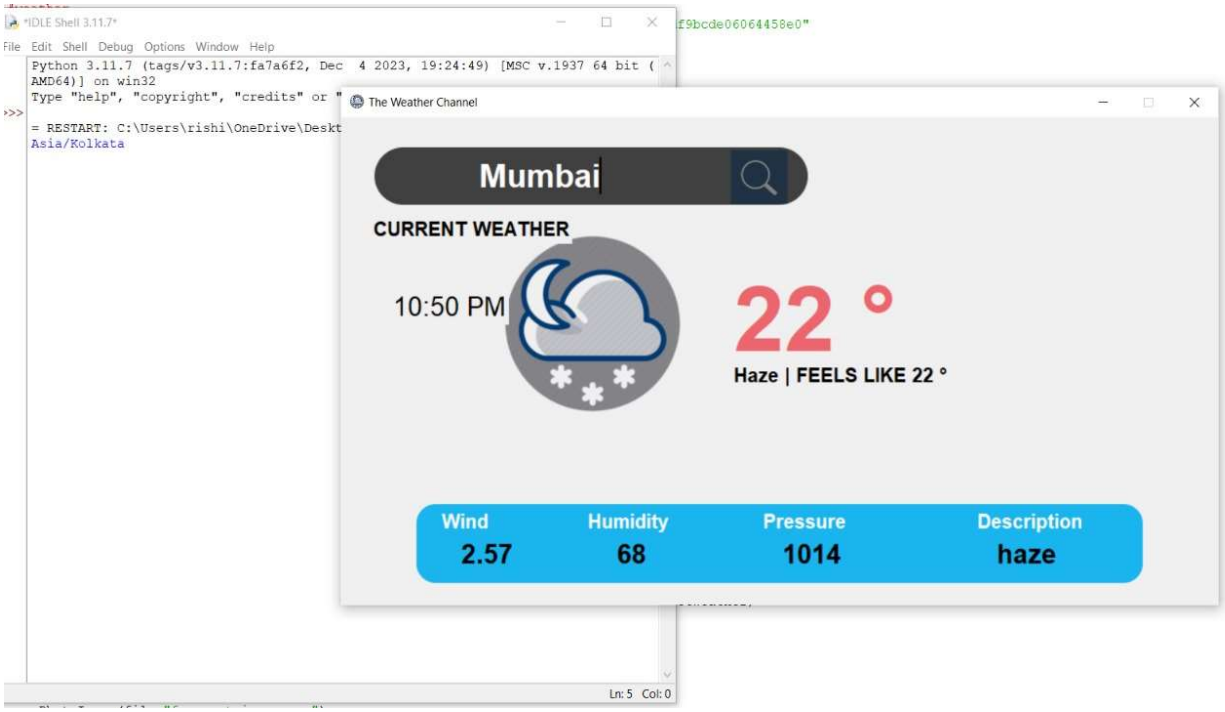
```
from tkinter import * import
tkinter as tk
from geopy.geocoders import Nominatim from
tkinter import ttk,messagebox
from timezonefinder import TimezoneFinder from
datetime import datetime
import requests import
pytz

root = Tk()
root.title("The Weather Channel")
root.geometry("900x500+300+200")
root.resizable(False, False)

def getWeather(): try:
        city=textfield.get()

        geolocator= Nominatim(user_agent="geoapiExercises") location=
        geolocator.geocode(city) obj=TimezoneFinder()
        result = obj.timezone_at(lng=location.longitude,lat=location.latitude)
        print(result)

        home = pytz.timezone(result) local_time =
        datetime.now(home)
        current_time = local_time.strftime("%I:%M %p")
        clock.config(text=current_time) name.config(text="CURRENT
        WEATHER")

        #weather

api="https://api.openweathermap.org/data/2.5/weather?q="+city+"&
appid=48b43465bfec45bdf9bcde06064458e0"

        json_data = requests.get(api).json() condition =
        json_data['weather'][0]['main']
        description = json_data['weather'][0]['description']
```

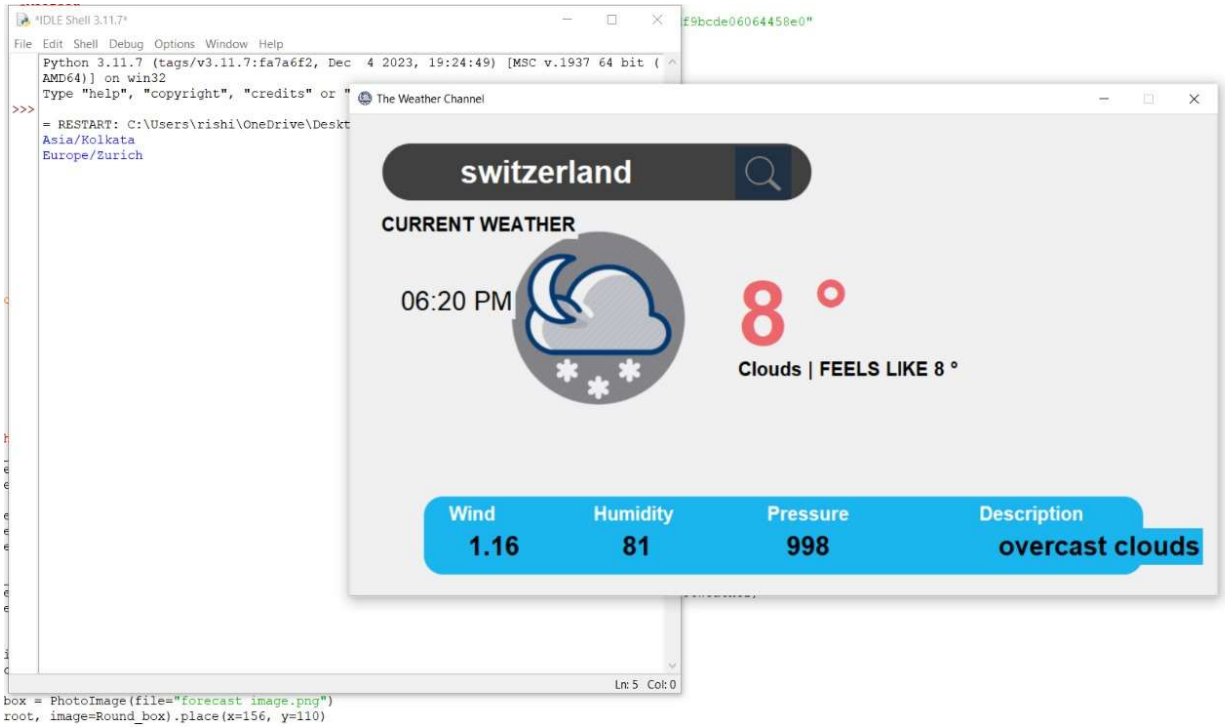
```
            temp =int(json_data['main']['temp']-273.15) pressure =
            json_data['main']['pressure'] humidity =
            json_data['main']['humidity'] wind =
            json_data['wind']['speed']

            t.config(text=(temp,"°"))
            c.config(text=(condition,"|","FEELS","LIKE",temp,"°"))

            W.config(text=wind)
            H.config(text=humidity)
            D.config(text=description)
            P.config(text=pressure)

        except Exception as e:
                    messagebox.showerror("Weather  App","Invalid
Entry!!")


#Search box
Search_image = PhotoImage(file="Copy of search.png") myimage =
Label(image=Search_image) myimage.place(x=20, y=20)

textfield = tk.Entry(root, justify='center', width=17, font=('poppins', 25, 'bold'),
bg="#404040", border=0, fg="white")
textfield.place(x=50, y=40) textfield.focus()

Search_icon = PhotoImage(file="Copy of search_icon.png") myimage_icon =
Button(image=Search_icon, borderwidth=0, cursor="hand2",
bg="#203243",command=getWeather) myimage_icon.place(x=400, y=34)
```

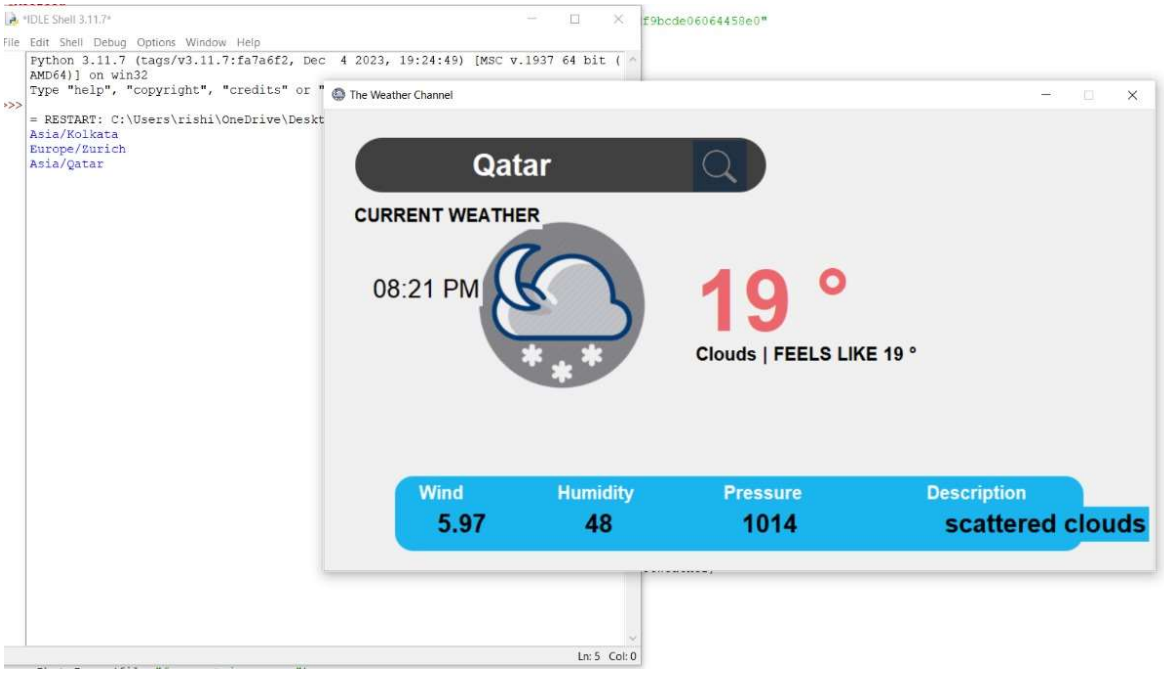
```
#logo
image_icon = PhotoImage(file="forecast image.png") root.iconphoto(False,
image_icon)

Round_box = PhotoImage(file="forecast image.png") Label(root,
image=Round_box).place(x=156,  y=110)

#Bottom Box
Frame_image=PhotoImage(file="Copy of box.png")
frame_myimage=Label(image=Frame_image)
frame_myimage.pack(padx=5,pady=5,side=BOTTOM)

#time name=Label(root,font=("arial",15,"bold"))
name.place(x=30,y=100)
clock=Label(root,font=("Helvetica",20))
clock.place(x=50,y=175)


# label
label1 = Label(root, text="Wind", font=("Helvetica",15,'bold'),fg="white",bg="#1ab5ef")
label1.place(x=100, y=400)

label2 = Label(root, text="Humidity", font=("Helvetica", 15,'bold'),
fg="white",bg="#1ab5ef")
label2.place(x=250,  y=400)

label3 = Label(root, text="Pressure", font=("Helvetica", 15,'bold'),
fg="white",bg="#1ab5ef")
label3.place(x=430,  y=400)

label4 = Label(root, text="Description", font=("Helvetica", 15,'bold'),
fg="white",bg="#1ab5ef")
label4.place(x=650,  y=400)

t=Label(font=("arial",70,"bold"),fg="#ee666d")
t.place(x=400,y=150) c=Label(font=("arial",15,'bold'))
```

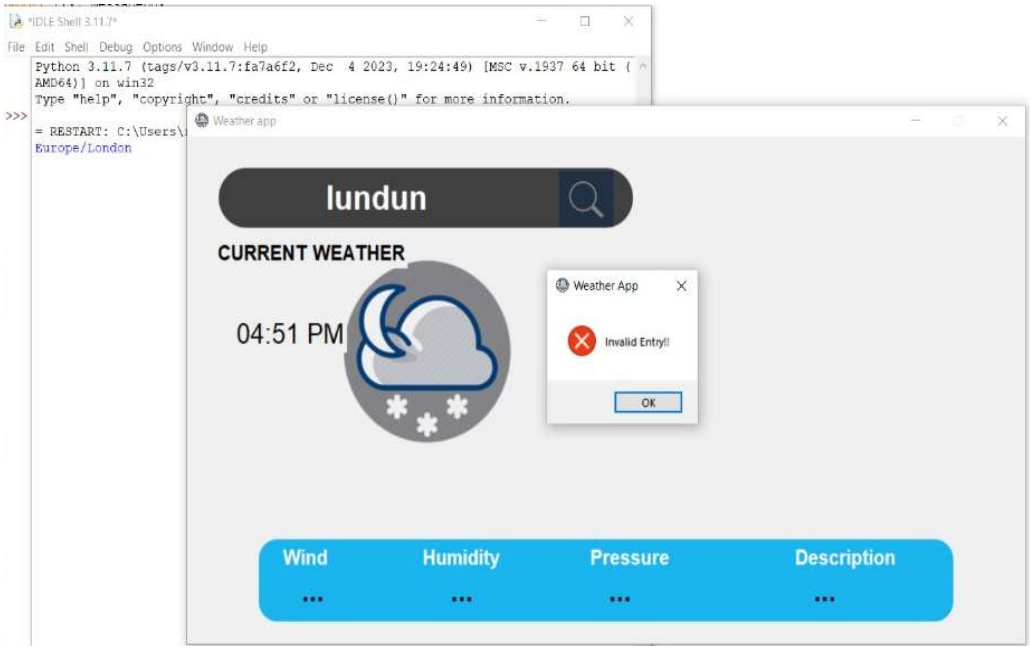
```
c.place(x=400,y=250)

W=Label(text="...",font=("arial",20,"bold"),bg="#1ab5ef")
W.place(x=120,y=430) H=Label(text="...",font=("arial",20,"bold"),bg="#1ab5ef")
H.place(x=280,y=430) P=Label(text="...",font=("arial",20,"bold"),bg="#1ab5ef")
P.place(x=450,y=430) D=Label(text="...",font=("arial",20,"bold"),bg="#1ab5ef")
D.place(x=670,y=430)




root.mainloop()
```

Final outputs






IDLE Shell 3.11.7
File Edit Shell Debug Options Window Help
Python 3.11.7 (tags/v3.11.7:fa7a6f2, Dec 4 2023, 19:24:49) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or
>>>
= RESTART: C:\Users\rishi\OneDrive\Deskt
Asia/Kolkata
Europe/Zurich
Asia/Qatar
Ln: 5 Col: 0
The Weather Channel
Qatar
CURRENT WEATHER
08:21 PM
19 °
Clouds | FEELS LIKE 19 °
Wind
5.97
Humidity
48
Pressure
1014
Description
scattered clouds


IDLE Shell 3.11.7
File Edit Shell Debug Options Window Help
Python 3.11.7 (tags/v3.11.7:fa7a6f2, Dec 4 2023, 19:24:49) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\
Europe/London
Weather app
lundun
CURRENT WEATHER
04:51 PM
Weather App
Invalid Entry!!
OK
Wind
...
Humidity
...
Pressure
...
Description
...

CONCLUSION

In conclusion, the development of this weather application using Python has been a rewarding endeavor, offering valuable insights into both programming and meteorology. Through the integration of various APIs and libraries, we have created a versatile tool capable of delivering accurate and timely weather information to users worldwide.

The Python programming language provided a robust foundation for this project, enabling efficient data manipulation, API communication, and user interface design. Leveraging libraries such as Requests, TimeZoneFinder and Tkinter allowed us to streamline the development process and create a seamless user experience.

Overall, this project has been an enriching learning experience, showcasing the power and versatility of Python in creating practical, real-world applications. By harnessing the capabilities of programming and meteorology, we have developed a valuable tool that empowers users to make informed decisions and stay informed about the ever-changing weather conditions.