A Mini Project Report On

PYTHON PROGRAMMING - A SIMPLE WEATHER APP

Submitted for partial fulfillment of the requirement for the award of degree of

BACHELOR OF BUSINESS ADMINISTRATION IN ARTIFICIAL INTELLIGENCE

BY

Jashan Chaudhary	**25030422054**
Ansh Sahani	**25030422016**
Aarav Sahni	**25030422004**

UNDER THE GUIDANCE OF
MR. DAWA LEPCHA

SYMBIOSIS ARTIFICIAL INTELLIGENCE INSTITUTE
Symbiosis International University Near Lupin Research Park, Gram: Lavale, Tal:, Mulshi, Maharashtra 412115

SYMBIOSIS ARTIFICIAL INTELLIGENCE INSTITUTE

Symbiosis International University Near Lupin Research Park, Gram:
Lavale, Tal:, Mulshi, Maharashtra 412115

CERTIFICATE

This is to certify that the project work entitled -"**PYTHON PROGRAMMING - A SIMPLE WEATHER APP"** that is submitted by -

Jashan Chaudhary	**25030422054**
Ansh Sahani	**25030422016**
Aarav Sahni	**25030422004**

in partial fulfillment for the award of degree of Bachelor of Business Administration in Artificial Intelligence is a record of bonafide work carried out by them. The results embodied in this report have been verified and found satisfactory.

Faculty in charge,
Mr .Dava Lepcha
Professor

DECLARATION

We **Jashan Chaudhary**, **Ansh Sahani**, **Aarav Sahni** here by dadqdawd declare that the report of the Mini Project work entitled "**PYTHON PROGRAMMING - A SIMPLE WEATHER APP** "which is being submitted to the SYMBIOSIS ARTIFICIAL INTELLIGENCE INSTITUTE PUNE, in partial fulfillment of the requirement for the award of Degree of Bachelor of Business Administration in Artificial Intelligence, is bonafide report of the work carried out by us. The material contained in this report has not been submitted to any University or Institution for the award of any degree.

Place: SAII, PUNE

Date: Nov 4, 2025

(Jashan Chaudhary - 25030422054)

(Ansh Sahani - 25030422016)

(Aarav Sahni - 25030422004)

ACKNOWLEDGEMENT

We would like to sincerely thank Professor Dawa Lepcha for his constant support and guidance throughout this project. His valuable insights and encouragement helped us stay motivated and focused.

This project allowed us to explore API integration and GUI development using Tkinter, which helped strengthen our technical and creative skills. We truly appreciate his patience and constructive feedback, which guided us in improving the quality and presentation of our work.

Finally, we express thanks to all those who have helped us to complete this project successfully. Furthermore, we would like to thank our family and friends for their moral support and encouragement.

(Jashan Chaudhary - 25030422054)

(Ansh Sahani - 25030422016)

(Aarav Sahni - 25030422004)

ABSTRACT

Simple Weather App is a Python desktop application built with the Tkinter library and the OpenWeatherMap API to fetch weather data of any city in real-time. It shows the essential details like the temperature and the general weather conditions in a simple, interactive way. The app features a dark theme and a centered layout which makes the text readable and the user experience more pleasant.

The application shows how Python can be used to integrate an API into a GUI, thus functionality is combined with design. Users can get the weather data of a city instantly by typing the city name in the input box. Error handling is there to give the user clear messages if the input is wrong or there is no connection to the internet. The interface becomes smooth and users get involved because of the presence of interactive- button hover effects plus the minimalistic style.

This is a great example to show how python can be used for creating real-life applications. It depicts the skills of the developer in GUI, API, and event-driven programming. The app demonstrates the fact that with just simple tools, one can create meaningful and efficient software solutions. Essentially, the Simple Weather App is not only a handy tool for fetching weather information, but also a stepping stone towards mastering modern Python application development.

TABLE OF CONTENTS

INTRODUCTION

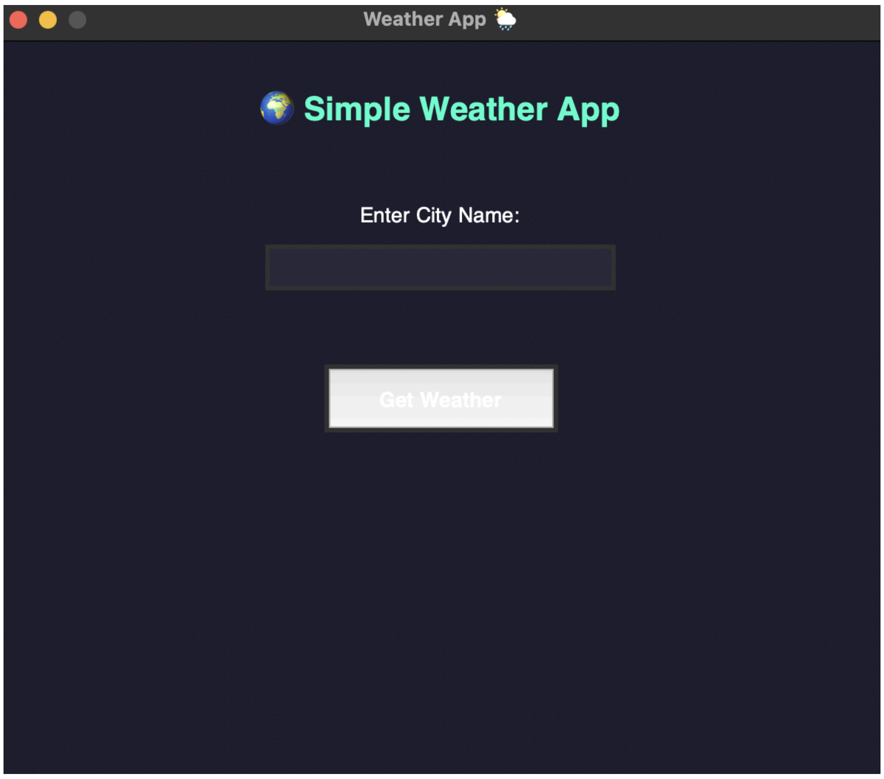
Simple Weather App is a tool that aims to deliver in the quickest possible way up-to-date weather information to the users This is a Python-based application that makes use of the Tkinter library and hence is capable of establishing a connection with OpenWeatherMap API for fetching the weather details such as temperature and general conditions for any city globally.

The first point of the project is not only to provide basic functionality but also to serve as a teaching tool for the practical use of Python when it comes to the integration of APIs and graphical user interfaces. The application also sees a chance here to depict different concepts like error handling, event-driven programming, and design aesthetics in desktop application development.

The reason for building such an app is primarily concerned with the creation of a minimalist, distraction-free application that can still manage to provide essential weather updates without the user having to go through a complex website or mobile app. A feature of this system is that it concentrates on the content being simple, functional, and user-friendly so that in no time anybody can find out what they want.

PROBLEM STATEMENT

Usually, getting fresh weather information involves the use of websites or mobile apps which can be quite confusing or heavy in terms of data. As a result, there is a demand for a convenient means of accessing the most basic weather information without having to deal with any kind of distractions. The Simple Weather App solves the problem by offering a small desktop application that can be used to access the latest and instant weather updates for any city. This app is very simple and easy to use, as it only shows the most necessary weather data, and thus, it can be considered as a perfect and efficient tool for the users' daily routine.



(Clean and Basic UI)

METHODOLOGY

The Simple Weather App is structured around combining clean data retrieval with a user-friendly interface. The app performs a local set of operations: it takes user input, gets data from an API, processes the data and then shows the results to the user.

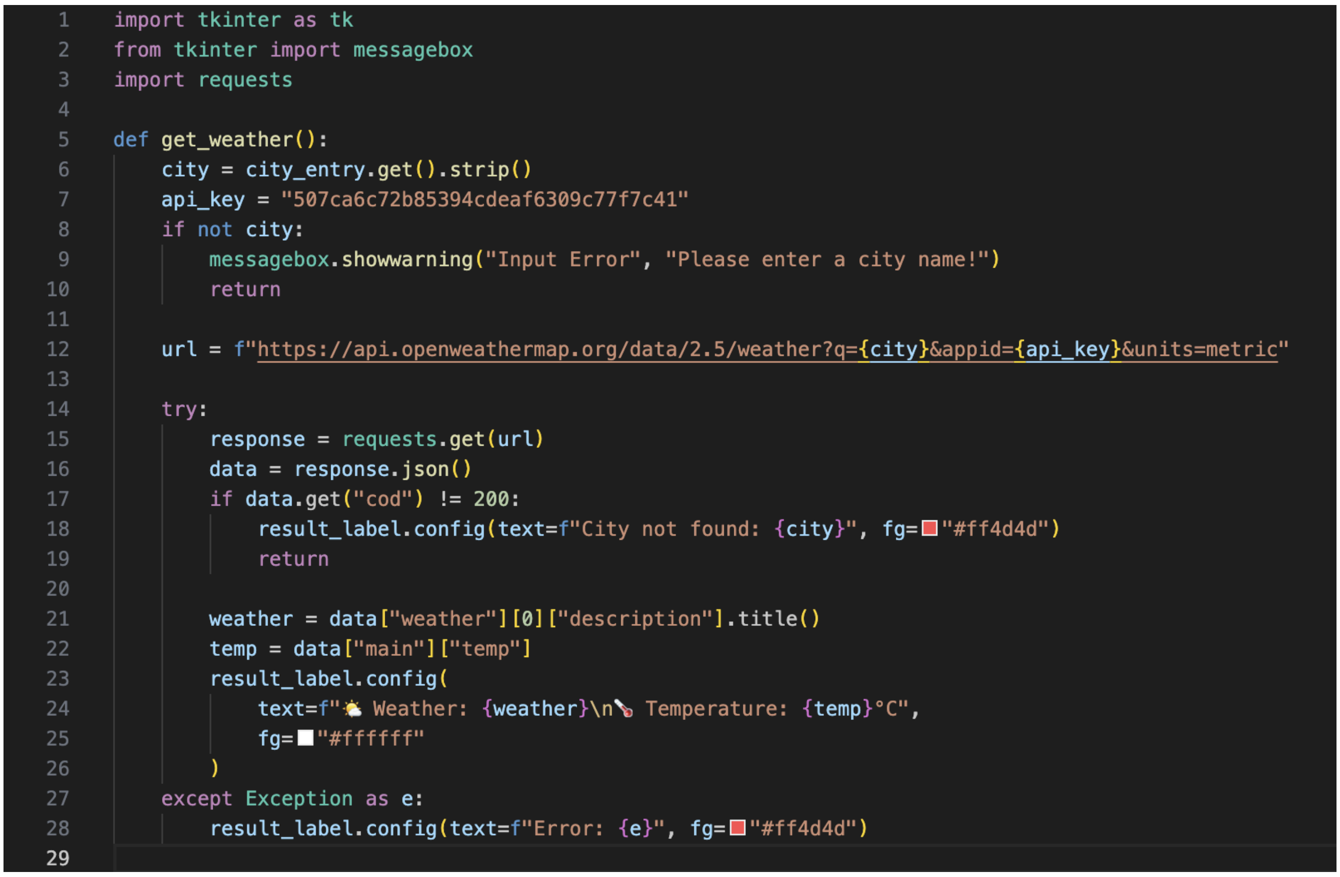
The project is implemented with Python, The Tkinter library is the tool that was used to build a graphical user interface (GUI) that can be easily used by customers via input fields and buttons. Also, the Requests library is the tool that is responsible for managing HTTP requests and responses to/from the OpenWeatherMap API, which is the source of live weather data.

Once the city name is typed by a user and the “Get Weather” button is clicked, the software formulates a request to the OpenWeatherMap API. After that, the information is shown to the user in a Tkinter window.

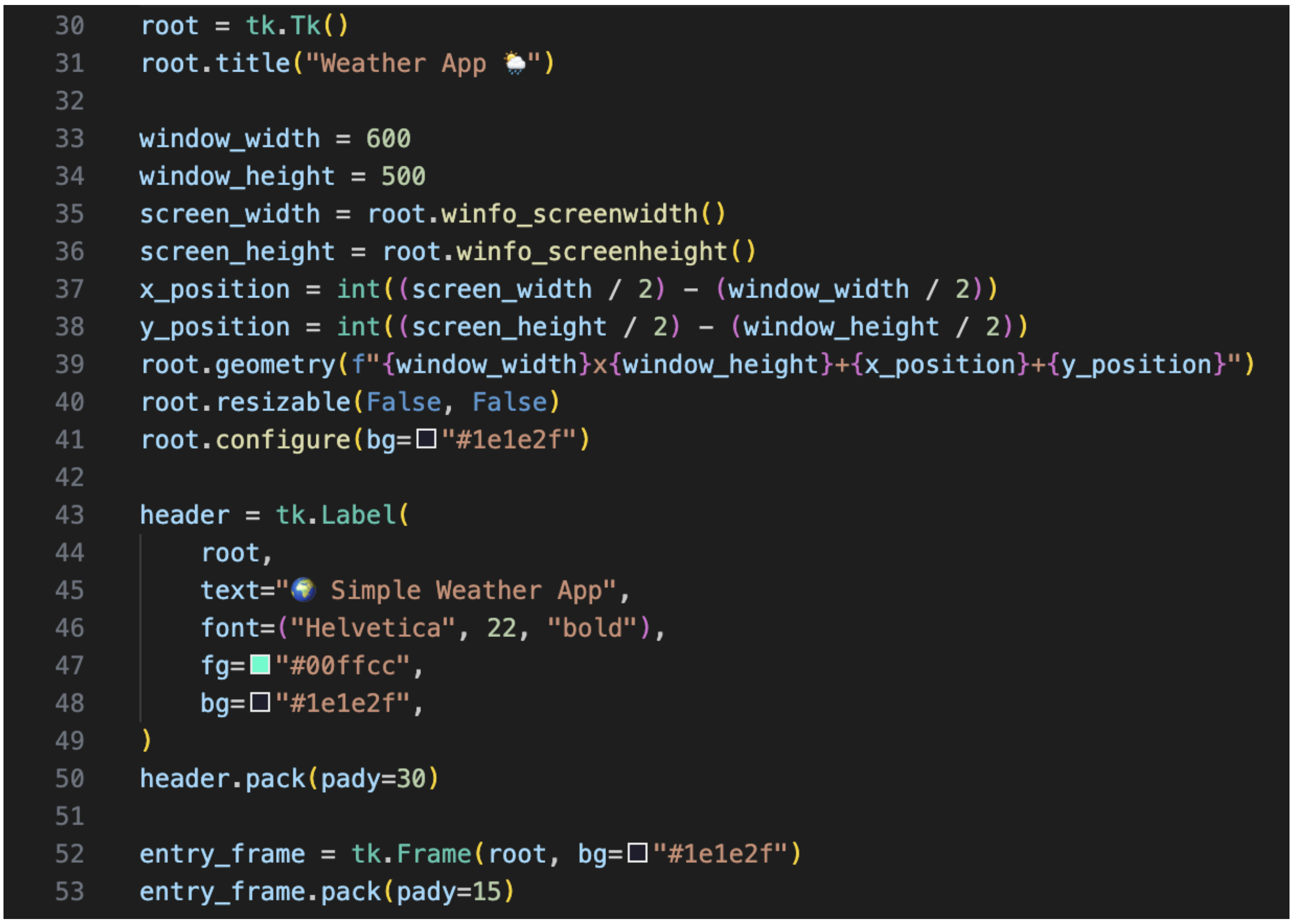
The strategy also features error handling that can recognize invalid city names as well as UI design improvements, such as hover effects and window position at the center, which gives users more pleasure. In addition, this mix of API linkage, GUI creation, and on-demand data handling is the main idea concept that the Simple Weather App was built on.

IMPLEMENTATION

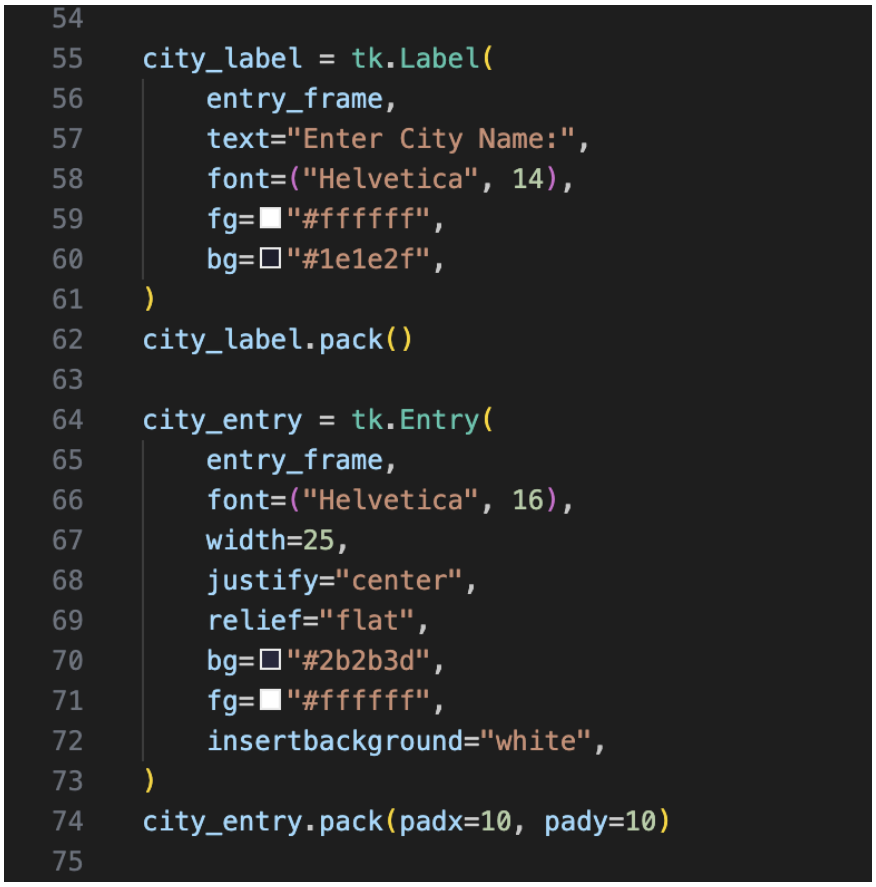
The Simple Weather App was implemented entirely in Python using the Tkinter library for the graphical interface and the Requests library for API communication. The development and testing were carried out on a standard desktop environment using VS Code and Python 3.x. No additional hardware components were required, as the project runs purely on software. Screenshots of the code and execution are provided below to illustrate the setup and functionality.



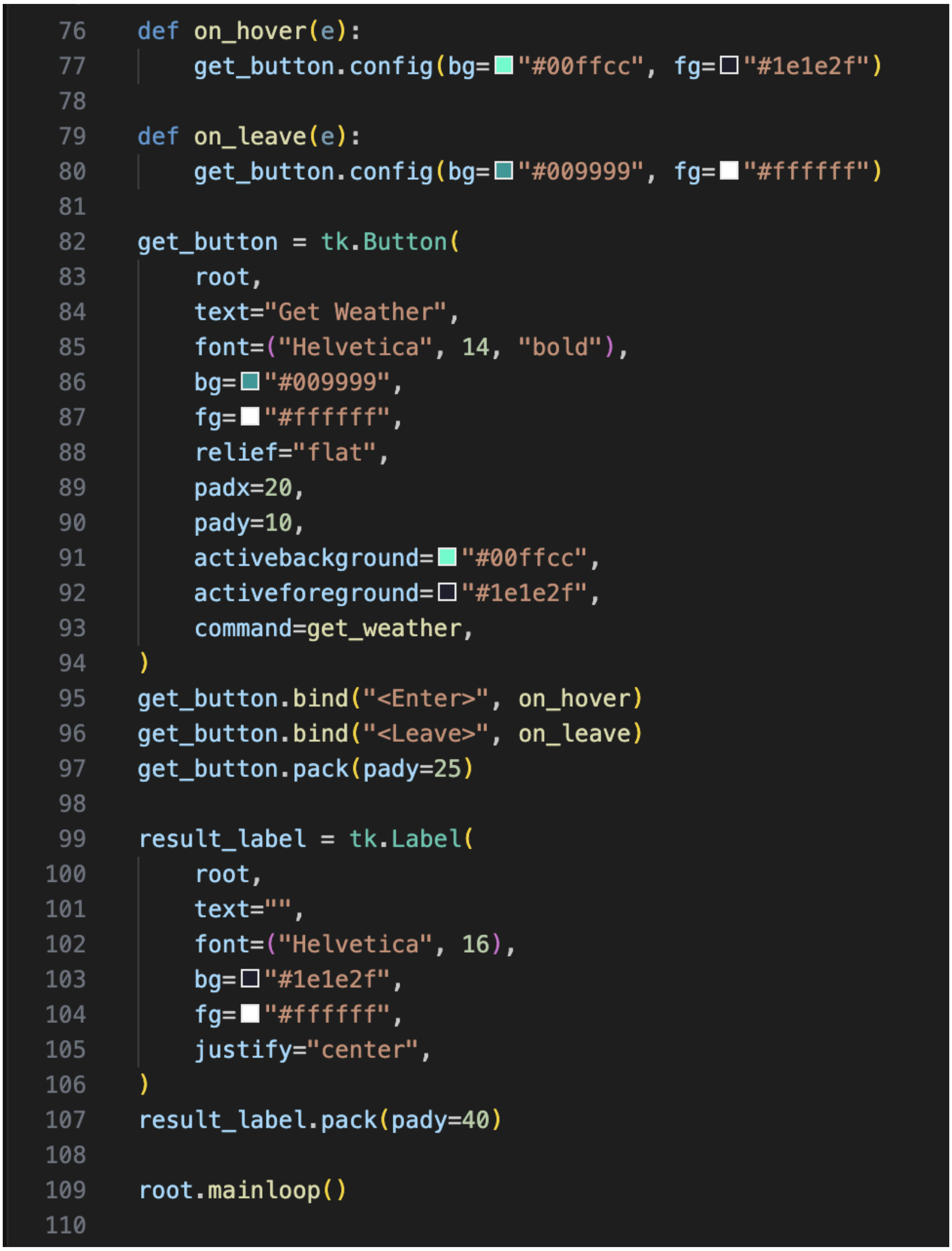
```python
import tkinter as tk
from tkinter import messagebox
import requests

def get_weather():
    city = city_entry.get().strip()
    api_key = "507ca6c72b85394cdeaf6309c77f7c41"
    if not city:
        messagebox.showwarning("Input Error", "Please enter a city name!")
        return

    url = f"https://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}&units=metric"

    try:
        response = requests.get(url)
        data = response.json()
        if data.get("cod") != 200:
            result_label.config(text=f"City not found: {city}", fg="#ff4d4d")
            return

        weather = data["weather"][0]["description"].title()
        temp = data["main"]["temp"]
        result_label.config(
            text=f"⛅ Weather: {weather}\n🌡 Temperature: {temp}°C",
            fg="#ffffff"
        )
    except Exception as e:
        result_label.config(text=f"Error: {e}", fg="#ff4d4d")

```



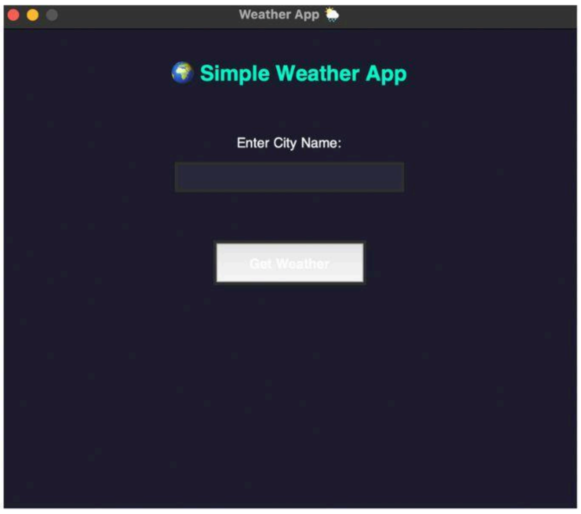
```python
root = tk.Tk()
root.title("Weather App 🌦️")

window_width = 600
window_height = 500
screen_width = root.winfo_screenwidth()
screen_height = root.winfo_screenheight()
x_position = int((screen_width / 2) - (window_width / 2))
y_position = int((screen_height / 2) - (window_height / 2))
root.geometry(f"{window_width}x{window_height}+{x_position}+{y_position}")
root.resizable(False, False)
root.configure(bg="#1e1e2f")

header = tk.Label(
    root,
    text="🌍 Simple Weather App",
    font=("Helvetica", 22, "bold"),
    fg="#00ffcc",
    bg="#1e1e2f",
)
header.pack(pady=30)

entry_frame = tk.Frame(root, bg="#1e1e2f")
entry_frame.pack(pady=15)
```



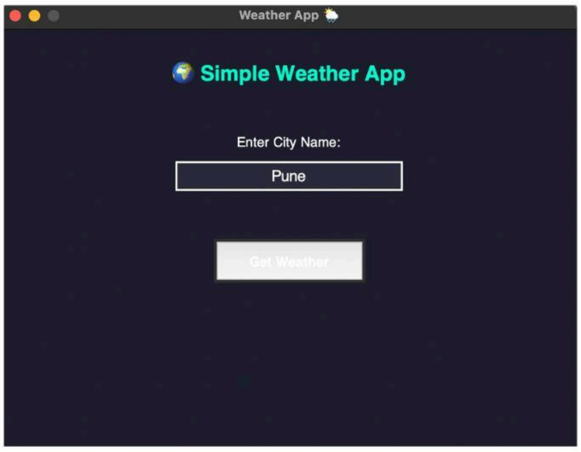
```python

city_label = tk.Label(
    entry_frame,
    text="Enter City Name:",
    font=("Helvetica", 14),
    fg="#ffffff",
    bg="#1e1e2f",
)
city_label.pack()

city_entry = tk.Entry(
    entry_frame,
    font=("Helvetica", 16),
    width=25,
    justify="center",
    relief="flat",
    bg="#2b2b3d",
    fg="#ffffff",
    insertbackground="white",
)
city_entry.pack(padx=10, pady=10)

```


```python
def on_hover(e):
    get_button.config(bg="#00ffcc", fg="#1e1e2f")

def on_leave(e):
    get_button.config(bg="#009999", fg="#ffffff")

get_button = tk.Button(
    root,
    text="Get Weather",
    font=("Helvetica", 14, "bold"),
    bg="#009999",
    fg="#ffffff",
    relief="flat",
    padx=20,
    pady=10,
    activebackground="#00ffcc",
    activeforeground="#1e1e2f",
    command=get_weather,
)
get_button.bind("<Enter>", on_hover)
get_button.bind("<Leave>", on_leave)
get_button.pack(pady=25)

result_label = tk.Label(
    root,
    text="",
    font=("Helvetica", 16),
    bg="#1e1e2f",
    fg="#ffffff",
    justify="center",
)
result_label.pack(pady=40)

root.mainloop()
```

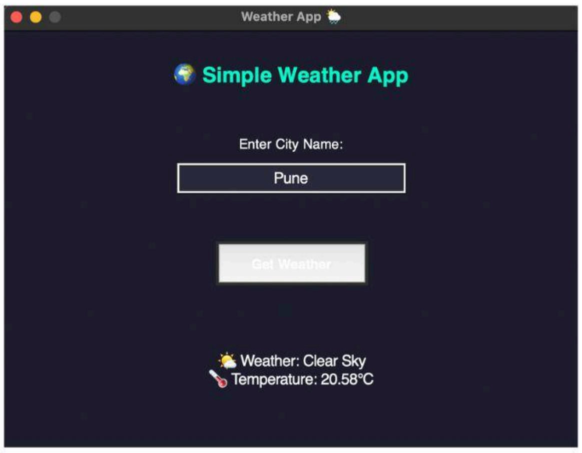
RESULTS



1.) Open Simple Weather App



2.) Enter city name



```
🌤 Weather: Clear Sky
🌡 Temperature: 20.58°C
```

3.) Get Weather results

APPLICATIONS

Simple Weather App can be used in many different contexts - in and outside the classroom - and can be quite helpful. The app may be utilized by users who want to obtain instantly current weather conditions for any city without the need of the web browser or using cumbersome mobile apps. Apart from that, it is a learning platform for students and developers, where they can grasp concepts like API integration, GUI development, and event-driven programming in Python through a basic project.

CONCLUSIONS AND FUTURE WORKS

The Simple Weather App is an excellent example of a tool that provides weather details accurately and in real-time via a straightforward and responsive user interface. The combination of Python, Tkinter, and OpenWeatherMap API is a good illustration of how external data can be a source for a user-friendly desktop application in an efficient way. This undertaking has been instrumental in learning the concepts of programming such as API handling, GUI design, and error management.

This application could be developed further by adding features such as the humidity level, wind speed, and weather forecast for several days. The user interface can be made more attractive by using trendy design elements or animations, which will also enhance the user experience. With these additions, the app would become more attractive to users and its practical value would increase.

REFERENCES

1.)Python Software
2.)OpenWeather. *OpenWeatherMap API*. Available at:
https://openweathermap.org/api
3.)Python Tkinter - *Graphical User Interface Programming*.