Python Project Report On Weather-forecast

Meghnad Saha Institute of Technology (MSIT), Kolkata

CSE,2nd Year students

***~~~~~~Project Contributor's: ~~~~~***

Yash Raj Singh (ROLL=17)

Amartya Pandey (ROLL=23)

Suvam Gupta (ROLL=24)

Aman Raj (ROLL=32)

Acknowledgement

This is a simple GUI based project build in python along with incorporation of RDBMS & the Open Weather Map API to draw the live data into our project.

We appointed

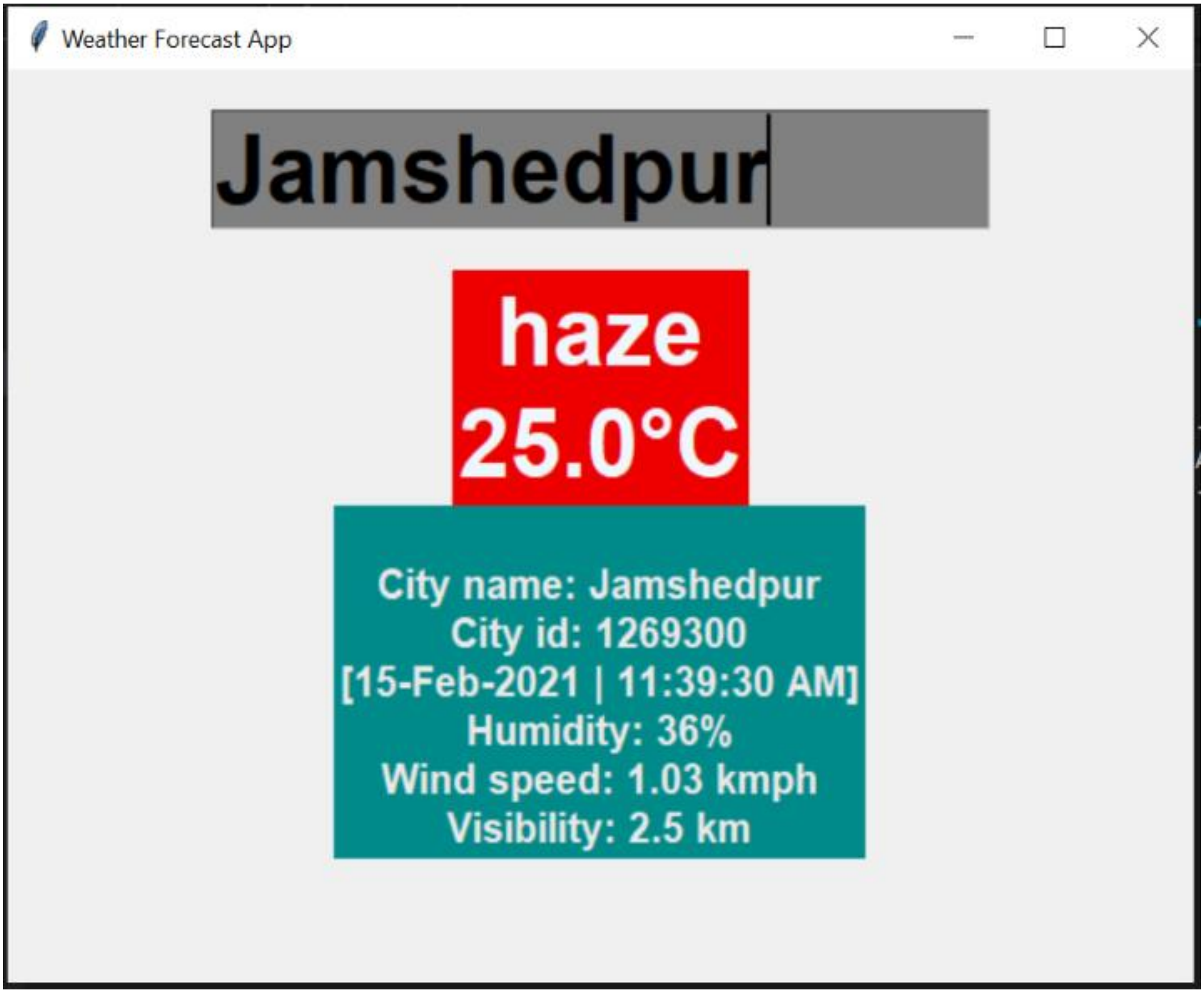
Yash Raj Singh

*as the head of our group member and we decided thoroughly how to implement as there are unprecedented ways to do a project then we all finalized that we would accomplish the project on time via use of Application Programming Interface and storing in a database using **sqlite3***

Project Description(Report)

Weather Forecast project is a python based **GUI** project with usage of few library (which encompasses certain functions) that are to be exported by using the pip install command to fetch the utility value of the any library, here we used Request , Json, Sqlite3 , Datetime & Tkinter library where Request is used to fetch the Hypertext Transfer Protocol fetched from the internet, Json that replaces skeleton html to key value pair and datetime library as the name suggest to exhibit real time in data, Sqlite3 to store in database, Tkinter for GUI support.

API used is the main heart of this project as it acts as an access point to the database such that any real time data can be drawn on immediate basis via the help of Api key.



Jamshedpur

haze
25.0°C

```
City name: Jamshedpur
City id: 1269300
[15-Feb-2021 | 11:39:30 AM]
Humidity: 36%
Wind speed: 1.03 kmph
Visibility: 2.5 km
```

Our project is saved in GITHUB

yashrajyash/Weather_Forecast (github.com)

This project is dived into 3 python files: -

- ★ show_weather.py (main)
- ★ fetch.py
- ★ sequel.py

show_weather.py is the main file which is responsible for showing output. It imports custom python files named fetch.py, sequel.py, datetime & Tkinter library, you can run this file in your system by: -

```
python show_weather.py (for windows)
```

```
python3 show_weather.py (for mac and Linux users)
```

fetch.py has function **fetch_data()** which returns desired city weather data in json format, here we have used OpenWeatherMap API and request module.

sequel.py has **create_table(), insert_into_table() and show_database()** functions related to SQL and it will store weather data to **weather.db** database file which is available in the root directory here we have used sqlite3 library.

To see the database run terminal in projects directory and hit command: -

```
python sequel.py
```

Code for show_weather.py

```
from fetch import fetch_data
from datetime import datetime
from sequel import *
import tkinter as tk



def get_weather(root):


    create_table()


    city = textField.get()
    api_data = fetch_data(city)


    if api_data['cod'] == '404':


        print('\n```````````````````````````````````````')
        print(city + ' ' + api_data['message'])
        print('```````````````````````````````````````')
```

```
    else:

        weather_desc = str(api_data['weather'][0]['description'])
        temp_city = str(float(api_data['main']['temp']) - 273.15)[:4] + '°C'
        humid = str(api_data['main']['humidity']) + '%'
        wind_spd = str(api_data['wind']['speed']) + ' kmph'
        city_name = str(api_data['name'])
        city_id = str(api_data['id'])
        date_time = '[' + str(datetime.now().strftime("%d-%b-%Y | %I:%M %p")) +
']'
        visibility = str(float(api_data['visibility'])/1000)[:4] + ' km'

        insert_into_table(city_id, city_name, date_time, temp_city, weather_desc,
wind_spd, humid, visibility)

        print('\n\n--------------------------------------------------------------
----------------------')
        print('Weather stats for -> {} | City-id : {} | {}'.format(city_name,
city_id, date_time))
        print('------------------------------------------------------------------
------------------\n')

        print('Current Temperature   :  {}'.format(temp_city))
        print('Weather Discription   :  {}'.format(weather_desc))
        print('Wind Speed            :  {}'.format(wind_spd))
        print('Humidity              :  {}'.format(humid))
        print('Visibility            :  {}\n'.format(visibility))
        final_info = weather_desc + '\n' + temp_city

        final_data = '\nCity name: ' + city_name + '\nCity id: ' + city_id + '\n'
+ date_time + '\nHumidity: ' + humid + '\nWind speed: ' + wind_spd +
'\nVisibility: ' + visibility



        label1.config(text = final_info)

        label2.config(text = final_data)




# GUI

root = tk.Tk()



root.geometry('600x450')



root.title('Weather Forecast App')



f = ("poppins", 15, "bold" )

t = ("poppins", 35, "bold")



textField = tk.Entry(root, width = 15, font = t, bg = 'grey')

textField.pack(pady = 20)

textField.focus()

textField.bind('<Return>', get_weather)
```

```
label1 = tk.Label(root, font=t, bg = 'red2', fg = 'alice blue')
label1.pack()
label2 = tk.Label(root, font=f, bg = 'cyan4', fg = 'snow2')
label2.pack()

root.mainloop()
```

Code for fetch.py

```python
from requests import get
import json


def fetch_data(city_name):

    api_key = 'cbb6b8a7a93022bfe48120a12e239d34'
    api_url = 'https://api.openweathermap.org/data/2.5/weather?q=' + city_name + '&appid=' + api_key
    content = get(api_url)
    return content.json()


def main():


    city = input('Enter city name : ').strip().capitalize()
    fetch_data(city)


if __name__ == '__main__':


    main()
```

Code for sequel.py

```
import sqlite3 as sql


def create_table():

    conn = sql.connect('weather.db')
    cur = conn.cursor()

    cur.execute("CREATE TABLE IF NOT EXISTS weather_details \
                (city_id TEXT, \
                city_name TEXT, \
                date_time TEXT, \
                temp_city TEXT, \
                weather_desc TEXT, \
                wind_spd TEXT, \
                humid TEXT, \
                visibility TEXT, \
                PRIMARY KEY (city_id, date_time, temp_city) \
                    );")

    conn.close()
```

```
def insert_into_table(cityId, cityName, dateTime, tempCity, weatherDesc, windSpd,
humid, visibility):


    conn = sql.connect('weather.db')

    cur = conn.cursor()



    cur.execute("INSERT INTO weather_details \

                (city_id, city_name, date_time, temp_city, weather_desc, wind_spd,
humid, visibility) \

                VALUES (?,?,?,?,?,?,?,?);",(cityId, cityName, dateTime, tempCity,
weatherDesc, windSpd, humid, visibility))



    conn.commit()



    conn.close()




def show_database():


    conn = sql.connect('weather.db')

    cur = conn.cursor()



    cur.execute("SELECT * FROM weather_details")

    rows = cur.fetchall()



    print('\n------------------------------  <UPDATED_DATABASE>  ----------------
----------------\n')
```

```
    for row in rows:
        print(row)


    conn.close()



def main():


    create_table()
    insert_into_table(cityId = '', cityName = '', dateTime = '', tempCity = '',
weatherDesc = '', windSpd = '', humid = '', visibility = '')
    show_database()


if __name__ == '__main__':


    check = 123


    if check == 404:
        main()
    else:
        show_database()
```

Contribution Details

Yash Raj Singh	Works on database, terminal output of main file, merge team's pull request or fixing merge conflicts & update readme file
Amartya Pandey	Works on API and parsing json data, managing the report book & fixing bugs while exporting functions.
Suvam Gupta	Analyze project, divide the project in 3 parts & creating the project structure.
Aman Raj	Woks on GUI, layout design of the main file & sending queries to database.