Project Report: Weather App1. Introduction:The Weather App is a software application designed to fetch real-time weather data from an online API and present it to the user in a user-friendly manner. The app can determine the user's location or accept a manually inputted location to provide accurate and up-to-date weather information.2. Objective:The main objective of the Weather App is to provide users with a convenient and accessible way to obtain current weather conditions for any location. The app aims to fetch data from a reliable weather API, process the information, and present it in an easy-to-understand format for the user.3. Features:a. Location Detection:The app utilizes the device's geolocation capabilities to determine the user's current location automatically.Users can also input a specific location manually if they want to check the weather for a different area.b. API Integration:Integration with a reputable weather API (e.g., OpenWeatherMap, Weatherbit, etc.) to obtain accurate and real-time weather data.The app handles API requests efficiently, ensuring a seamless experience for the user.c. User Interface:A clean and intuitive user interface that displays relevant weather information such as temperature, humidity, wind speed, and more.The UI adapts to various screen sizes for a responsive design.d. Forecast:Provides a short-term or extended weather forecast, depending on the user's preference.Graphical representation of temperature variations over time.e. User Settings:Allows users to customize units (e.g., Celsius or Fahrenheit) and other preferences.Option to save favorite locations for quick access.f. Error Handling:Gracefully handles errors, such as network issues or API failures, and provides informative error messages to the user.

4. Technology Stack:Programming Language: JavaScript (React for the front end, Node.js for the back end).API: OpenWeatherMap API (or any other chosen weather API).Styling: CSS for styling and responsive design.Storage: Local storage for saving user preferences and favorite locations.

5. Implementation:The front end is developed using React to create a dynamic and interactive user interface.The back end, built with Node.js, handles API requests, processes data, and communicates with the front end.Asynchronous programming is utilized to fetch and update weather data in real-time.6. Challenges Faced:Ensuring accurate location detection and handling location-related permissions.Addressing potential API rate limits and optimizing API requests.Implementing a responsive design for a seamless user experience across devices.

7. Future Enhancements:Integration with additional weather APIs for redundancy and enhanced data accuracy.Inclusion of weather alerts and notifications.Historical weather data display and analysis.

8. Conclusion:The Weather App project successfully accomplishes its goal of providing users with real-time and accurate weather information. The integration of user-friendly features and a robust technology stack ensures a reliable and enjoyable user experience. Future enhancements can further elevate the app's capabilities, making it a valuable tool for users seeking up-to-date weather data.

Code :

#Using command line do "pip install requests"

#Internet connection is required for the program to run

#The maximum capacity is 60 runs/min so try staying below it

from tkinter import \*

import tkinter.font as font

import requests

import json

root=Tk()

root.title("Weather Conditions")

myfont= font.Font(family='Helvetica',size=20,weight='bold')

myfontbig= font.Font(family='Helvetica',size=60,weight='bold')

myfontsmall= font.Font(family='Helvetica',size=15)

frame=LabelFrame(root)

frame.pack(padx=10,pady=10)

def submit():

    top=Toplevel()

    frame2=LabelFrame(top)

    frame2.pack(padx=10,pady=10)

    city=entry.get()

    entry.delete(0,END)

    complete\_url=base\_url+"appid="+api\_key+"&q="+city

    response=requests.get(complete\_url)

    x=response.json()

    cityog=x['name']

    country=x['sys']['country']

    current\_temp=round(x['main']['temp']-273.15,1)

    feels\_like=round(x['main']['feels\_like']-273.15,1)

    weather=x['weather'][0]['main']

    min\_temp=round(x['main']['temp\_min']-273.15,1)

    max\_temp=round(x['main']['temp\_max']-273.15,1)

    label1=Label(frame2,text=cityog+","+country)

    label1['font']=myfont

    label1.grid(column=1,row=0)

    label2=Label(frame2,text=str(current\_temp)+" °C")

    label2['font']=myfontbig

    label2.grid(column=1,row=1)

    frame3=LabelFrame(frame2)

    frame3.grid(column=1,row=2,pady=5)

    label3=Label(frame3,text="feels like "+str(feels\_like)+" °C",anchor=W)

    label3['font']=myfontsmall

    label3.grid(column=1,row=3,columnspan=2)

    label4=Label(frame3,text=weather)

    label4['font']=myfont

    label4.grid(column=1,row=2)

    label5=Label(frame3,text="min/max temp: "+str(min\_temp)+"°/"+str(max\_temp)+"°")

    label5['font']=myfontsmall

    label5.grid(column=1,row=4)

api\_key="3e048c850e7bcecc34437519ce82156a"

base\_url="https://api.openweathermap.org/data/2.5/weather?"

Label(frame,text="Enter City Name:",font=myfont).grid(column=1,row=0)

entry=Entry(frame)

entry['font']=myfont

entry.grid(column=0,row=1,columnspan=3,padx=10,pady=10,ipadx=50,ipady=5)

button=Button(frame,text="Submit",command=submit)

button['font']=myfont

button.grid(column=1,row=2,ipadx=20,pady=10)

root.mainloop()