Report On SMS Weather Alert Notification using Twilio

INT 346
(Robot Process Automation Basics)

Submitted By:- **Arihant Jain(Member 1)**

Reg No.: 12018540

Section.: KO262

Roll No.: RKO262A64

Member 2:- Tushar Bharti

Reg No. – 12017437

Section – KO262

Roll No. – RKO262A14

Submitted To :- **Md. Imran Hussain (26819)**

LOVELY PROFESSIONAL UNIVERSITY

Abstract: -

Weather apps are extremely helpful in a world where weather is becoming increasingly unpredictable. SMS weather alerts play an important role during disasters. Most people don't think about planning for a natural disaster until it's too late. SMS emergency alerts can prevent this from happening. These alerts ensure that friends and family aren't left in the dark during emergencies. Mobile devices make it so much easier for anyone to receive emergency updates. People can get alerts about current weather, temperature and more ,in their current location.

Users also gain access to the services that they need to stay safe. Most services prefer to use SMS to send out weather text alerts and status updates. Mobile devices are often faster and more reliable than radio and TV stations. They work even when cable or power is out. And, people can use them to receive and share text message alerts with their contact list.

Introduction: -

This project is about sending custom SMS messages containing weather data based on user's location.

The task is to create an automation bot that detects user's location automatically and fetch local weather data that can be transmitted via SMS messages.

We have used 'Automation Anywhere' platform to create a bot and coded our script in python. 'Geocoder' API has been used to detect user's current location, and 'OpenWeatherMap' API to fetch weather data and, 'Twilio SMS Messaging' API to push notifications on their mobile phones.

Used Libraries :-

Request – The requests module allows us to send HTTP requests using Python. The HTTP requests returns a Response data(content, encoding, status, etc).

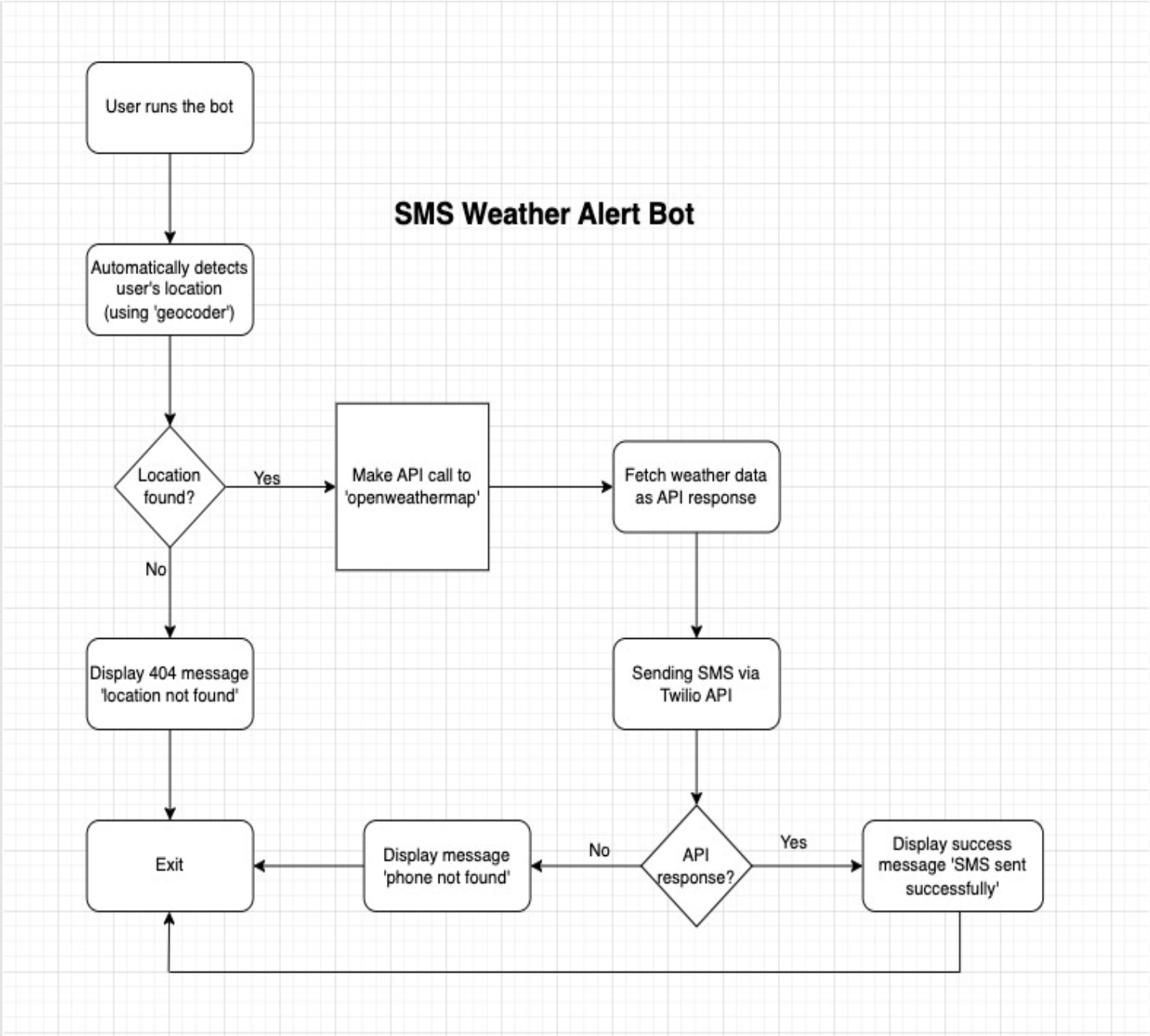
JSON – JSON is a syntax for storing and exchanging data.json is text, written with JavaScript object notation. Python has a built-in package json, which can be used to work with JSON data**.**

OS – This module provides a portable way of using operating system dependent functionality. The *os* and *os.path* modules include many functions to interact with the file system.

Geocoder – Geocoder is a simple and consistent geocoding library written in Python. Dealing with multiple different geocoding provider such as Google, Bing, OSM & many more has never been easier.

Twilio – The Twilio Python Helper Library makes it easy to interact with the Twilio API from our Python application. It is used for sending SMS on mobile phones

Flowchart:-


SMS Weather Alert Bot
User runs the bot
Automatically detects user's location (using 'geocoder')
Location found?
Yes
No
Make API call to 'openweathermap'
Fetch weather data as API response
Display 404 message 'location not found'
Sending SMS via Twilio API
Exit
Display message 'phone not found'
No
API response?
Yes
Display success message 'SMS sent successfully'

Working of Project:-

(i) Created a bot on Automation Anywhere.

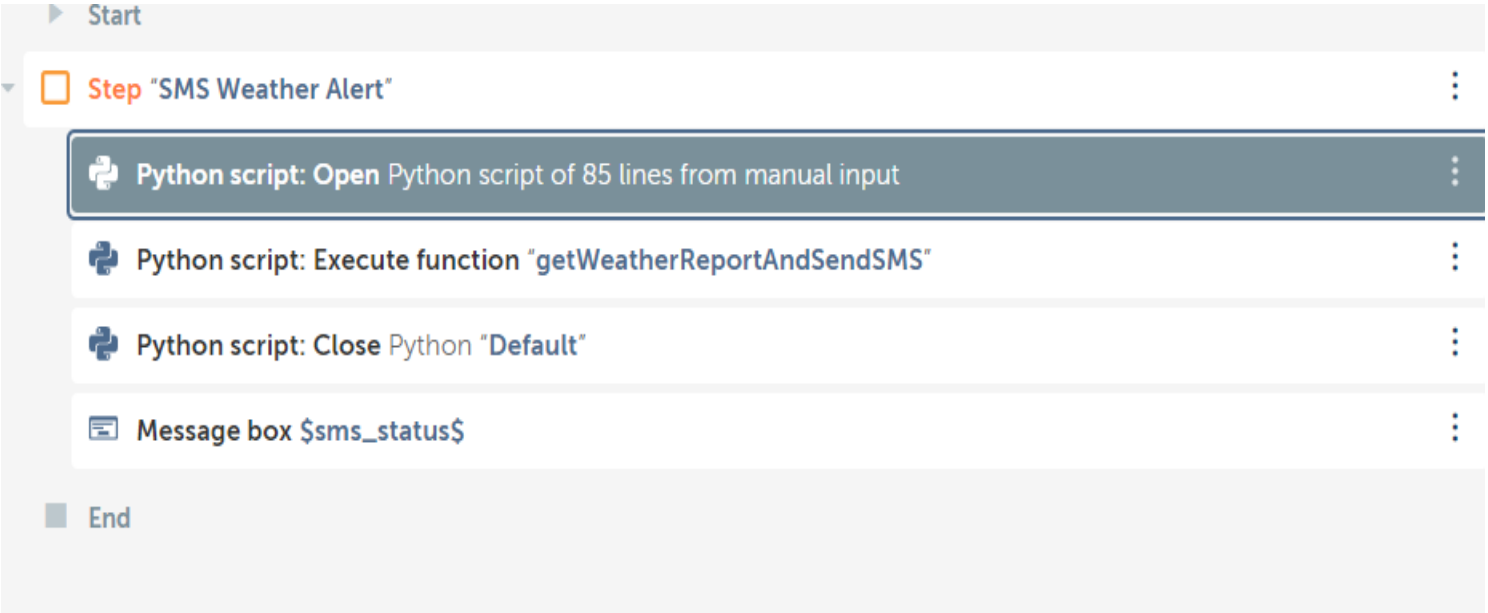
(ii) Added a step on the working panel.

(iii) Created an executable python script via ‘Open’ action.

(iv) Added manual python code that contains a function to fetch weather data and sent it via ‘Twilio’ API.

(v) Next, executed that python function via ‘Execute’ action.

(vi) Finally, closed the python session and displayed the ‘success’ message in message box.



What's inside Python Script:

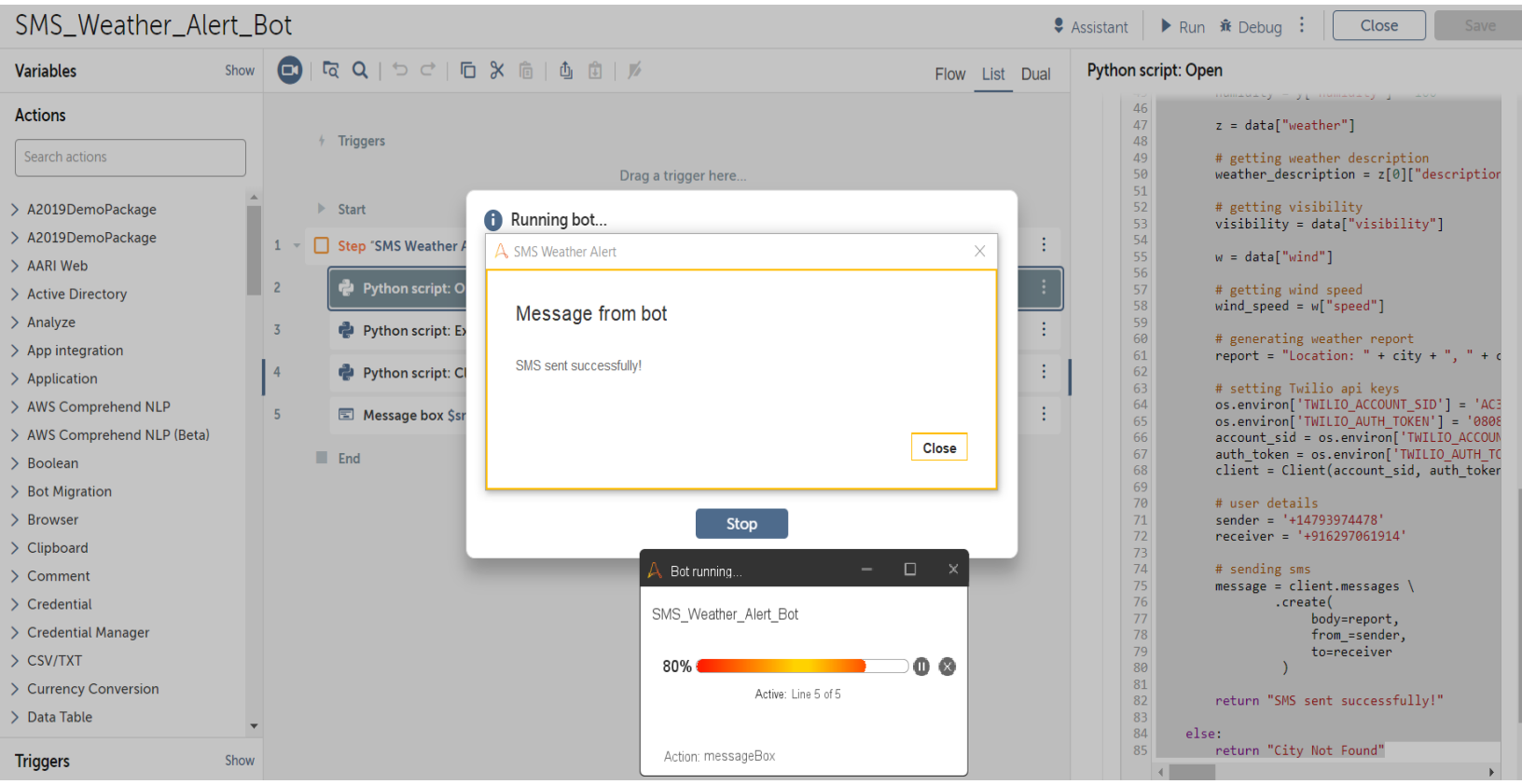
Python script: Open

```python
# importing lib
import requests, json
import os
import geocoder
from twilio.rest import Client

def getWeatherReportAndSendSMS():

    # weather api key
    api_key = "4463ff2ecafda4bdbd0866bff63de1f1"

    base_url = "http://api.openweathermap.org/data/2.5/weather?"

    # get current loaction
    g = geocoder.ip('me')

    # calling api
    complete_url = base_url + "appid=" + api_key + "&lat=" + str(g.latlng[0]) + "&lon=" + str(g.latlng[1])
    response = requests.get(complete_url)

    # getting data
    data = response.json()

    if data["cod"] != "404":

        # get city name
        city = data["name"]

        c = data["sys"]

        # get country code
        country_code = c["country"]

        y = data["main"]

        # getting current temperature
        temperature = y["temp"]
        temperature = temperature - 273.15
        temperature = round(temperature, 2)

```

Python script: Open

```
        z = data["weather"]

        # getting weather description
        weather_description = z[0]["description"]

        # getting visibility
        visibility = data["visibility"]

        w = data["wind"]

        # getting wind speed
        wind_speed = w["speed"]

        # generating weather report
        report = "Location: " + city + ", " + country_code + "\nTemperature (in celsius): " + str(temperature) + "\nAtmospheric Pressur

        # setting Twilio api keys
        os.environ['TWILIO_ACCOUNT_SID'] = 'AC3cc7844764fea46af7d6871f0e3d8487'
        os.environ['TWILIO_AUTH_TOKEN'] = '080811cab06cfa014e289a137e2392a0'
        account_sid = os.environ['TWILIO_ACCOUNT_SID']
        auth_token = os.environ['TWILIO_AUTH_TOKEN']
        client = Client(account_sid, auth_token)

        # user details
        sender = '+14793974478'
        receiver = '+916297061914'

        # sending sms
        message = client.messages \
                .create(
                     body=report,
                     from_=sender,
                     to=receiver
                 )

        return "SMS sent successfully!"

    else:
        return "City Not Found"
```

Output Screenshots:



Conclusion:

This is to conclude that we have successfully implemented the use of Weather, Location and SMS APIs in our SMS Alert Bot. Through this project, we have learnt how to use APIs and capture their responses and to integrate those responses on the Automation Anywhere platform. Also, we have gained the experience to write custom python scripts according to the need. This project has helped us grasping the concepts and architecture of APIs.

Future Scope

After building this SMS weather bot, we realized that we can customize or expand on it with other data or add even more weather-related advice to share with the user. Additionally, if we are interested in upgrading our OpenWeather API access, we can even get historical and forecast data that may be interesting to include in our weather bot application.

References

https://www.twilio.com/blog/build-weather-chatbot-sms-python-flask

https://www.bandwidth.com/blog/building-a-sms-weather-and-image-bot/

https://openweathermap.org/current

https://www.youtube.com/watch?v=SXsaB9TUfkk