Lesson 3 Demo 9

AutoGen Weather Assistant

Objective: To develop an AutoGen-based agent that fetches real-time weather data, provides insights, integrates external APIs, and handles errors gracefully

You are building an intelligent assistant that fetches real-time weather data for a given city and provides useful insights. This requires seamless integration with an external weather API (OpenWeatherMap) while ensuring the AutoGen agent can dynamically process and act on external data in real time. This agent should:

1. Fetch real-time weather data from an external weather API (for example, OpenWeatherMap) based on user input (city name)
2. Provide meaningful insights based on the weather data (for example, current temperature, weather conditions like clear or rainy)
3. Demonstrate integration of external APIs with AutoGen, showcasing the flexibility of agents in dynamically interacting with APIs and making decisions based on real-time data
4. Handle errors and edge cases gracefully, ensuring the agent can respond with appropriate messages if the API is down or if the input data is invalid (for example, an incorrect city name)

Prerequisites:

1. Create a virtual environment
2. Activate the virtual environment
3. Install dependencies

Steps to be followed:

1. Setup the environment
2. Define the function to fetch weather data using the OpenWeatherMap API
3. Define the weather agent using AutoGen
4. Create Streamlit UI
5. Run the webapp

Step 1: Set up the environment

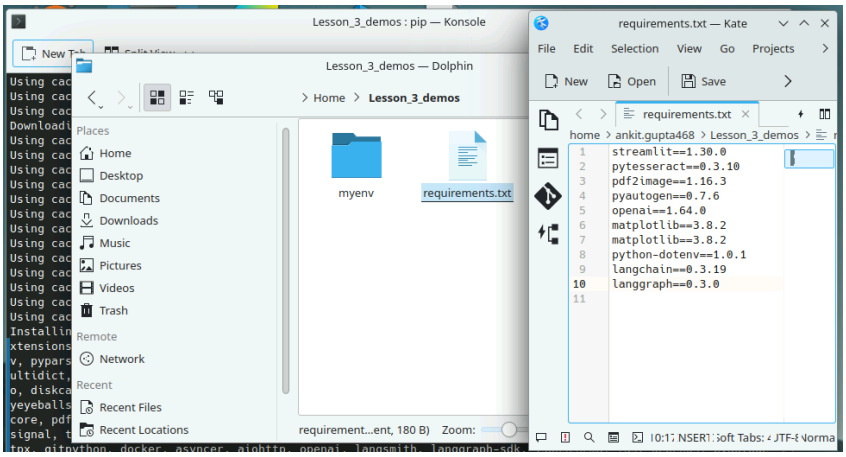
1.1 Open command prompt and go to the "**Lesson_3_demos**" folder (which we created in Demo_1) using the command given below:

mkdir Lesson_3_demos (not needed if the folder is already created in Demo1)
cd Lesson_3_demos

1.2 After this, activate the virtual environment using the command below:

python3 -m venv venv (not needed if the virtual env. is already created in Demo1)
source venv/bin/activate

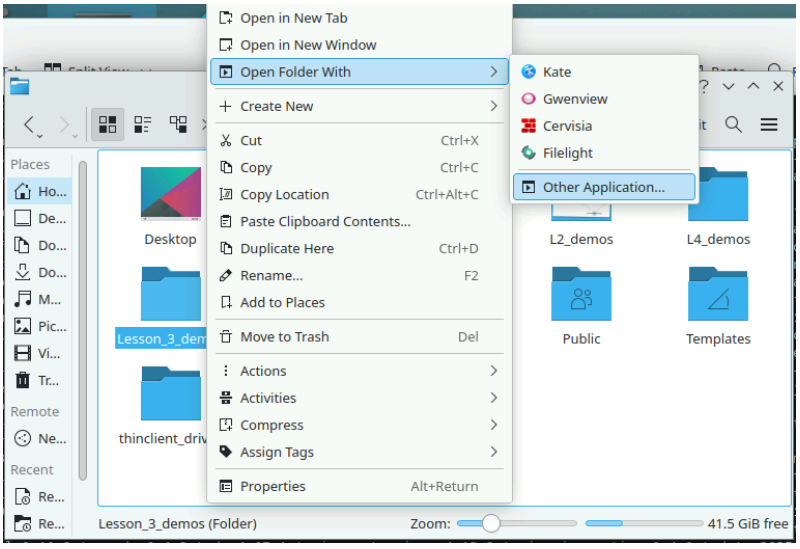
1.3 Now, create a `requirements.txt` file inside the folder with required libraries (not needed if already done in Demo1):



1.4 Install all the required libraries using the command below:

pip install -r requirements.txt (not needed if already done in Demo1)

1.5 Now, open the folder using VS Code editor:



1.6 After this, open a new Python file using the "**New File**" option and name it as "**Demo9**".

Step 2: Define the function to fetch weather data using the OpenWeatherMap API

2.1 The get_weather function takes a city and an API key as inputs and makes an API request to OpenWeatherMap using the requests.get function, and then it returns the weather data. If the request is successful, indicated by a status code of 200 (OK), it returns the weather data in JSON format. Otherwise, it returns an error message.

```
import streamlit as st
import requests
from autogen import AssistantAgent
from dotenv import load_dotenv
import os

# Load environment variables
load_dotenv()
weather_api_key = os.getenv("weather_api_key")

# Define the function to fetch weather data using the OpenWeatherMap API
def get_weather(city: str, api_key: str) -> str:
    api_key = "9e0c9a2d7478f9d535424f1778fc6a9d"
    url =
f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}&units=metric"
    response = requests.get(url)
    if response.status_code == 200:
        data = response.json()
        temperature = data['main']['temp']
        weather_description = data['weather'][0]['description']
        return f"The current temperature in {city} is {temperature}°C with
{weather_description}."
    else:
        return "Unable to fetch weather data."
```

Step 3: Define the weather agent using AutoGen

```
# Define the Weather Agent using AutoGen
weather_agent = AssistantAgent(
    name="Weather_Agent",
    llm_config={
        "config_list": [
            {
                "api_type": "azure",
                "azure_endpoint": "https://openai-api-management-gw.azure-
api.net/deployments/gpt-4o-mini/chat/completions?api-version=2023-12-01-preview",
                "api_version": "2023-12-01-preview",
                "api_key":
"2ABecnfxzhRg4M5D6pBKiqxXVhmGB2WvQ0aYKkbTCPsj0JLKsZPfJQQJ99BDAC77bzfXJ3w3AAABACOGi3sC",
                # "deployment_name": "gpt-4o-mini",  # corrected key name
                "model": "gpt-4o-mini",  # optional, but good to keep
            }
        ],
        "temperature": 0.7,
    },
    system_message="I can fetch weather data for any city.",
)
```

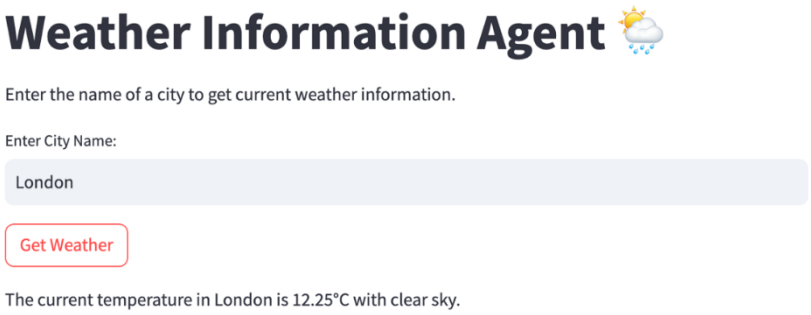
Step 4: Create Streamlit UI

```
# Streamlit UI
def main():
    st.title("Weather Information Agent")
    st.markdown("Enter the name of a city to get current weather information.")

    city = st.text_input("Enter City Name:")

    if city and st.button("Get Weather"):
        result = get_weather(city, weather_api_key)  # Fetch weather data

        # Display result
        st.write(result)

# Step 4: Run the Streamlit app
if __name__ == "__main__":
    main()
```

Step 5: Run the webapp

5.1 Save the file and then run the streamlit webapp from command prompt using the command given below:
streamlit run Demo9.py

Output:



By following the above-mentioned steps, you have successfully demonstrated how to integrate an external weather API with an AutoGen agent, enabling real-time, dynamic responses based on user inputs. By leveraging Streamlit for the user interface, the app allows users to easily retrieve weather information for any city by interacting with the AutoGen agent. This approach highlights the flexibility of AutoGen in interacting with external APIs, handling user queries, and providing actionable insights. The integration also showcases the simplicity of deploying the solution using Streamlit, making it accessible as a web application for broader use.

Appendix:

Step-by-step guide to setting up OpenWeatherMap API:

1. Register for an account on OpenWeatherMap

1.1 Visit the OpenWeatherMap website: Go to OpenWeatherMap.

1.2 Create an Account:

a) Click on the **Sign Up** button at the top right corner of the page
b) Fill in the required information, such as your email, password, and other details to create your account

2. Obtain your API key

5.2 Log In to Your Account: After registration, log in to your account

5.3 Navigate to the API Section:

a) Once logged in, go to the **API Keys** section by clicking on your username at the top right and selecting **My API Keys**.
b) Alternatively, go to the **API** page via the main menu and select the type of API you need.

5.4 Generate an API Key:

a) In the API Keys section, click the API Key button to generate a new API key. This will give you a unique API key.
b) You can name it for identification if you plan to use multiple keys for different projects.

6 Note down the API key

Once generated, copy the key (a long string of characters) and store it safely. This is what you'll use to authenticate your requests to the OpenWeatherMap API.