## **WeatherWise: Intelligent Weather Assistant - Full Code Explanation**

This Python program is an **interactive weather assistant** that:

* Fetches real-time weather using wttr.in API,
* Shows temperature and rain forecast charts using matplotlib,
* Understands natural language weather questions,
* Runs on the terminal using a menu-based interface.

### **🔹 Imported Libraries**

python

CopyEdit

import requests

import matplotlib.pyplot as plt

import pyinputplus as pyip

import os

* requests: Sends HTTP requests to fetch weather data from wttr.in.
* matplotlib.pyplot: Used to draw temperature and rainfall charts.
* pyinputplus: Handles user inputs safely (e.g., menu, integers).
* os: Used for system-level tasks (though not directly used here).

### **🖼️ welcome\_message() – Displays Introductory Text**

Prints a styled welcome banner in the terminal to introduce WeatherWise and guide users on how to interact with the program.

### **get\_weather\_data(location, forecast\_days=5)**

Fetches JSON weather data from the wttr.in API:

* Constructs the URL based on the city entered.
* Sends a GET request.
* Checks if weather and current condition data is present.
* Returns a structured dictionary with:  
  + Current weather,
  + Forecast for the next n days (up to 5).

It handles errors gracefully by printing a warning if something fails.

### **create\_temperature\_visualisation(weather\_data, output\_type='display')**

* Extracts max and min temperatures for upcoming days.
* Uses matplotlib to plot line graphs of temperature trends.
* Labels the chart with city name and displays it.
* Can also return the plot object (if output\_type != 'display').

### **☔ create\_precipitation\_visualisation(weather\_data, output\_type='display')**

* Extracts **chance of rain** from the 5th hourly forecast (around mid-day).
* Creates a **bar chart** of rain probabilities across days.
* Includes grid, labels, and title with city name.
* Displays or returns the plot.

### **parse\_weather\_question(question)**

Processes a user’s natural language question (e.g., “Will it rain tomorrow in Port Louis?”):

* Detects if the user is asking about **temperature** or **rain**.
* Checks if the question is about **today** or **tomorrow**.
* Extracts the **location** using in keyword.
* Returns a dictionary with weather type, time, and location.

Example:

python

CopyEdit

'Will it rain tomorrow in Curepipe?'

→ {'type': 'rain', 'time': 'tomorrow', 'location': 'Curepipe'}

**generate\_weather\_response(parsed\_question, weather\_data)**

* Uses the parsed question to extract either today's or tomorrow’s forecast.
* Returns a weather summary:  
  + If asking for **temperature** → Min & Max °C
  + If asking for **rain** → Chance of rain in %

### **main\_menu() – The Interactive Terminal UI**

This is the heart of the app. It displays a menu and waits for user selection:

python

CopyEdit

☁️ Fetch Weather (by City)

💬 Ask AI about Weather

📈 Show Temperature Chart

☔ Show Precipitation Chart

❌ Exit

**Menu Options in Detail:**

1. **Fetch Weather (by City)**
   * Asks for city and days.
   * Displays current weather in that city.

**Ask AI about Weather**

* + Lets user type a weather-related question.
  + Uses the AI parser to understand the question and respond smartly.

1. **Show Temperature Chart**
   * Shows line graph of min & max temps for next few days.
2. **Show Precipitation Chart**
   * Shows rain probability as bar chart.
3. **Exit**
   * Ends the application.

### **if \_\_name\_\_ == '\_\_main\_\_':**

This line makes sure that the program starts by calling main\_menu() **only** when this script is run directly, not when imported into another file.

## **Summary**

**WeatherWise** is a Python-based CLI app that:

* Uses real-time data from wttr.in,
* Shows beautiful data visualizations with matplotlib,
* Accepts both typed city inputs and natural language questions,
* Handles errors and user input robustly,
* Is modular and readable, making it easy to expand or adapt.