Weather App using OpenWeather API

Introduction:

This program uses the OpenWeather API to reach out to OpenWeather and gather weather information for cities in the United States. The following features are in the application:

- Temperature from any US city that the user inputs
- Temperature conversion from Kelvin to Fahrenheit and Celsius
- Data
    - Windspeed
    - Wind direction.
    - Humidity for the given location

Design:

I designed the app a while back to gather weather data from different cities in the U.S. In order to make this work, I have an API key from OpenWeather that allows me to get the weather data in a JSON format. It takes user inputs to enter the name of the city and the state code. The program adds then add the city, state code, and API key to the base URL and sends that request to OpenWeather.

After the users enter the city and state, you will see the complete URL, response from the site and the weather data. You can copy and paste the URL into the browser, and you will get the data in a JSON format. I then had to parse the data from the JSON file to get it in the correct outputs. I had to do temperature conversions because the temperature was in Kelvin and that required me to make the conversions to Fahrenheit and Celsius.

The hardest part was converting this from structured to object-oriented code. I converted all of the old if/elif statements to a WindConditions class. This conversion required me to create a conditions list that had the degrees that were matched to the cardinal directions. Once complete I had to create a function and use a for loop to loop across the list and match directions and degrees.

```
Enter name of city: Dallas
Enter state code: TX
https://api.openweathermap.org/data/2.5/weather?q=Dallas,TXUS&appid=ac1d69c1d98523be1dfd8ee6e45bd8e2
<Response [200]>
The temperature is, 88 degrees Fahrenheit, 31 degrees Celsius with humidity at 57% .
Winds are out of the southwest with windspeeds of 3.09 mph.
```

Conclusion:

Takeaways:

- I learned how to refactor old, structured code into objects.
- Don't bite off more than you can handle.
- Start earlier.

Best Features:

- Getting weather data in JSON format
- Getting the okay response from the website
- The temperature conversions
- The wind speed and directions

Short Comings:

- The site does have all of the U.S. cities.
    - The weather will be based off of the major city with the same name>
        - Ex: Alexandria, VA and Alexandria, LA
            - Alexandria, VA weather will be the response from OpenWeather.
- I don't think that it provides up-to-date temperature and windspeeds.

Choices:

- I would stick to what I decide in the beginning and not change midstream.
- At this stage for me, I should have refactored this from the beginning, instead of trying to do a more advanced project (still working on it)

Additional Features:

- Add conditions (sunny, cloudy, rain,) etc...
- Try another API to see if I can get more accurate information.
- Work on the state codes.