Iteration 3

Completing the Hourly Forecast Classes

In this phase, you'll build the core classes responsible for handling hourly forecast data in your Weather Forecast Application. These classes will allow the system to represent, convert, load, and manage forecast data that was previously written to CSV by the ForecastWorker.

You will complete two Python files:

- `hourly_forecast_class.py`
- `hourly_forecast_manager_class.py`

Each contains `TODO` comments to guide your work.

Your Task

Hourly Forecast Class

File: `hourly_forecast_class.py`

This class represents an hour of weather data.

You will:

- Write two helper functions for converting temperature between Fahrenheit and Celsius.
- Create a dictionary to map icons and emojis
- Create a daily forecast class
- Write the `__init__` method
- Implement a function to create an hourly forecast object from a dictionary that will:
    - Extract and format various metrics of weather
    - Handle cases where value or unit is missing.
    - Convert the temperature to the other unit.
    - Return an hourly forecast object using the cleaned and converted data.

Hourly Forecast Manager Class

File: `hourly_forecast_manager_class.py`

This class loads the CSV file that contains hourly forecasts and turns each row into an hourly forecast object.

You will:

- Implement the hourly forecast manager class with these responsibilities:
    - Store the CSV filename, the forecast generation time, and the forecasts.
    - Load the forecast data from the CSV
    - Convert each row into an hourly forecast object

- Handle errors during file loading and return `True` or `False` accordingly.
- Return the list of forecasts via a getter method.

Testing Your Code

You can test your code by running the main script. You should see a window similar to the previous iteration, but there will be an additionaly tab to show hourly forecasts.

What to Submit

Push your completed versions of `hourly_forecast_class.py` and `hourly_forecast_manager_class.py` to your GitHub repository.

Make sure to:

- Replace all TODO comments by completing the code.
- Use good syntax and style.
- Add meaningful commit messages.

Example commit message

```
git commit -m "Implement HourlyForecast and HourlyForecastManager for
Iteration 3"
```

Congratulations, be proud of yourself.