Jae Ammar

Professor Sabine Rosenberg

CART 351

Project 1 Description

For this project I wanted to experiment with how environmental data can be displayed, and designed so it can be felt. The goal was to blend technical coding practice with creativity. Instead of displaying air-quality numbers in a plain way, I wanted to make the data feel more alive and expressive. Using live information the the World Air Quality Index, I built an interactive Python interface where the terminal itself sort of becomes a stage. Interactivity is central. Users type a city name, and the program retrieves real-time data, labelling each station through a combination of textual categories and visual cues. Each with a color, an emoji, and a bar. The user searches cities, inspects individual stations by UID, and can select different calculations, choosing how to see the same data. Using API helpers, colorized output, ascii banners(via libraries) demonstrated the modular python design. The overall goal is to blend data, requests, input handling, exceptions with aesthetics. Making environmental data playful and clear.

"AQI Theater" shows the same observations in three modes – raw(0-500), normalized (0-100), and 'misrepresented' nonlinear version that exaggerates differences. This mode demonstrated how visual encoding choices can distort perception and invoke possible misinformation by data dramatization. "Fog Monster" mini-game makes a monster based off AQI. The monster's health and damage scale is based on local AQI. High pollution makes the boss stronger and more punishing; fighting it becomes a metaphor for pollution itself. Healing, attacks, and escape odds reflect on air quality. I also included an ASCII creature '*Norns'*, by the artist Shanaka Dias from the ASCII Art Archive. Libraries like pyfiglet, colorama, and art helped transform a plain terminal into a playful visual space that turns real environmental data into small interactive artwork that is informative, playful, and

thought-provoking. Users can also access station details by inputting the station's UID to see all the components behind it. Quick City Search allows you to input a city keyword, then displays all stations found with their UID, Lat, Long, and bar to determine air quality.

The expected outcomes for this project had two different important parts, the technical and aesthetic. Technically it had to use live data and terminal-based interaction to generate user-responsive dynamic outputs. Each function contributes to a smooth, self-contained code that runs in the command line and the user can navigate between modes, retrieve real time data, and visualize it through text animation, ASCII form, and color. From a design perspective, this project was difficult since you had to operate within constraints, the terminal was both the medium and the aesthetic frame. Having to work purely in a text based environment eliminates the conventional polished user interface design and replaces it with an interface that depends on rhythm, color, and typographic weight rather than imagery.

References & Resources

World Air Quality Index Project Team. (2024). *World Air Quality Index API documentation.* Retrieved from https://aqicn.org/api/.

Python Software Foundation. (2024). *Colorama documentation.* Retrieved from https://pypi.org/project/colorama/.

GeeksforGeeks. (2023). *Python ASCII art using PyFiglet module.* Retrieved from https://www.geeksforgeeks.org/python/python-ascii-art-using-pyfiglet-module/.

Grinberg, M. (2024, October 7). *The ultimate guide to error handling in Python.* Miguel Grinberg Blog. Retrieved from https://blog.miguelgrinberg.com/post/the-ultimate-guide-to-error-handling-in-python.

Zarzu, O. (2024, March 8). *Python requests best practices for data engineers.* Y42 Blog. Retrieved from https://www.y42.com/blog/python-requests-best-practices-for-data-engineers.