Project Write-Up: Alpha Stock Library

Sahil Patel

What compromises did you make due to time constraints?

- I wanted to add comments to core.py and utils.py for clarity and maintainability.
- Did not implement a more robust, persistent caching solution like Redis. Currently, the cache is in-memory and only lasts for the life of the program.
- The Alpha Vantage API allows for outputsize=full to retrieve the entire time series. Right now, the implementation uses the default (outputsize=compact), which only returns the most recent 100 entries. This limits the ability to look up older data. Ideally, I would add logic to inspect how far back the requested date is and use the appropriate outputsize parameter.

How would you approach versioning of this library?

I would follow Semantic, using a MAJOR.MINOR.PATCH scheme:

- Increment MAJOR for breaking changes,
- MINOR for backwards-compatible feature additions,
- PATCH for bug fixes and small improvements.

I set the current version to 0.1.0 as this library still likely needs more work before it can be released for public use.

How would we go about publishing this library?

To publish the library:

1. Package it using setuptools and wheel,
2. Use twine to upload it to PyPi

How would you design this if it was going to be a service rather than a library?

If building this as a service:

- I would expose the lookup, min, and max operations via a RESTful API (e.g., using FastAPI or Flask).
- The service would include a persistent cache (e.g., Redis) and a background job to prefetch or refresh data.
- Authentication (API keys or OAuth) would secure usage.
- This would allow multiple clients or users to access consistent data over time without managing state.

Please include any other comments about your implementation.

- The data is sorted by date only when _get_sorted_dates is called (currently only used by min/mix). Then, it gets cached.
- For the min and max operations, the relevant portion of the time series is sorted before applying the operation.

How much time did you spend on this exercise?

- ~20 minutes familiarizing myself with the Alpha Vantage API.
- ~20 minutes learning about setup.py and how to structure a Python package.
- ~1 hour implementing the solution.
- ~20 minutes adding documentation and completing this write-up.