Task Two

pseudobabble

October 23, 2022

Contents

1 Usage

```
cd task_two
python3.8 -m venv venv
source venv/bin/activate
pip install -r requirements.txt
python app.py &
curl localhost:8080/London
# {"city":"London","temperature":17.45,"timestamp":"Sun, 23 Oct 2022 12:33:24 GMT"}
```

2 Approach

Spent too long on Task One - should have spent the additional time here.

3 Technical Choices

3.1 Libraries

- Flask
  - Simple, familiar, fast development
- SqlAlchemy
  - Familiar, rich functionality, good ORM, uses Data Mapper pattern not Active Record
- `requests`
  - Widely used and familiar, good `http` client
- `pseudobabble/repository`
  - Simple persistence abstraction I wrote for convenience

3.2 Approach

Trying to keep it readable, with domain relevant naming, logic for different purposes kept segregated, configurable with dependecies.

Out of time for tests and making it pretty.

3.3 Architecture

- Service oriented architecture
- Try to keep to SOLID
- Stateless services, stateful models

3.3.1 `CityWeather` model

Fairly straightforward SqlAlchemy model, can tell callers if it is stale (domain rules).

3.3.2 `CityWeatherRepository`

Just a persistence wrapper for convenience.

3.3.3 `CityWeatherService`

Encapsulates the logic required to perform the task. Dependencies injected for testability, extensibility, etc.

3.3.4 `WeatherClient`

Simple requests Session wrapper. Improvements would be: API token from env, add more complex parameter parsing and passing. More defensive code (`KeyError`), and similar.

4 Questions

The requirements say:

> If a request is made to your API for a city where the weather data was not requested before, OR previously requested more than 4 hours in the past, you should fetch the weather data from the third party OpenWeatherMap API and store the data in your SQL table. Once the data is stored in your database, you should return this data to the user.
>
> If a request is made to your API for a city where the weather data was previously requested 2 hours or less in the past, you should return the response directly from the SQL table.

Does this mean that stale data is 2+ hours old, or 4+ hours old? There is a gap 2-4 hours where we will not get the data directly from the table, and we will also not update the data. 3hr old data will be returned directly from the OpenWeather API.

I have opted to say that 4+ hours is stale: stale data will trigger an update to persistence, and the new value will be shown to the user.