DEVGRID

Python Tech Test

Design and build a service that collects data from an Open Weather API and store it as a JSON data.

Specifications:

- Use of Python 3 is mandatory.
- Service API with following endpoints:
    - POST - Receives a user defined ID, collect weather data from Open Weather API and store:
        - The user defined ID (needs to be unique for each request)
        - Datetime of request
        - JSON data with:
            - City ID
            - Temperature in Celsius
            - Humidity
    - GET - Receives the user defined ID, returns with the percentage of the POST progress ID (collected cities completed) until the current moment.
- Async calls to 'Call for several city IDs' API to get weather information from the cities provided below at Appendix A.
- Free account of Open Weather API has a limit of 60 cities per minute. Service needs to get all cities provided respecting the mentioned limit.
- Create an account at Open Weather API to generate a token to call their API. This token is private information so it needs to be handled as a service configuration.
- Code needs to have more than 90% of test coverage.
- Open git repository (preferable in github).
- Include a Dockerfile to set up the environment and run the project.
- The solution should be replicated

Requirements:

- Any tool, database or framework can be used to help building the service
  - All decisions of using any tools, database or frameworks need to be explained.
- The service needs to run as a Docker application.

Tips:

- To request all the cities it should take a long time, be careful about timeouts.
- Remember to write a good README.md with the instructions necessary to run the application and the tests.

Deliverables:

- Link to the git repository
- Documentation:
  - Docker installation
  - How to run
  - How to test

Appendix A - Cities ID list

3439525, 3439781, 3440645, 3442098, 3442778, 3443341, 3442233, 3440781, 3441572, 3441575, 3443207, 3442546, 3441287, 3441242, 3441686, 3440639, 3441354, 3442057, 3442585, 3442727, 3439705, 3441890, 3443411, 3440054, 3441684, 3440711, 3440714, 3440696, 3441894, 3443173, 3441702, 3442007, 3441665, 3440963, 3443413, 3440033, 3440034, 3440571, 3443025, 3441243, 3440789, 3442568, 3443737, 3440771, 3440777, 3442597, 3442587, 3439749, 3441358, 3442980, 3442750, 3443352, 3442051, 3441442, 3442398, 3442163, 3443533, 3440942, 3442720, 3441273, 3442071, 3442105, 3442683, 3443030, 3441011, 3440925, 3440021, 3441292, 3480823, 3440379, 3442106, 3439696, 3440063, 3442231, 3442926, 3442050, 3440698, 3480819, 3442450, 3442584, 3443632, 3441122, 3441475, 3440791, 3480818, 3439780, 3443861, 3440780, 3442805, 7838849, 3440581, 3440830, 3443756, 3443758, 3443013, 3439590, 3439598, 3439619, 3439622, 3439652, 3439659, 3439661, 3439725, 3439748, 3439787, 3439831, 3439838, 3439902, 3440055, 3440076, 3440394, 3440400, 3440541, 3440554, 3440577, 3440580, 3440596, 3440653, 3440654, 3440684, 3440705, 3440747, 3440762, 3440879, 3440939, 3440985, 3441074, 3441114, 3441377, 3441476, 3441481, 3441483, 3441577, 3441659, 3441674, 3441803, 3441954, 3441988, 3442058, 3442138, 3442206, 3442221, 3442236, 3442238, 3442299, 3442716, 3442766, 3442803, 3442939, 3443061, 3443183, 3443256, 3443280, 3443289, 3443342, 3443356, 3443588, 3443631, 3443644, 3443697, 3443909, 3443928, 3443952, 3480812, 3480820, 3480822, 3480825.