Senior Software Engineer code challenge

Field data API

We would like you to build a REST API that manages fields and provide a weather history for a given field with a given boundary using a third partner API from OpenWeather Agro Monitoring.

Your API must be able to handle the two use cases described in the below sections.

Manage a field

As an API consumer, I'd like to be able to maintain my field through a REST API endpoint so that I can **C**reate, **R**etrieve, **U**pdate and **D**elete a field.

Acceptance criteria

- The API should accept all the operations through a single endpoint. i.e: `/fields`; `/fields/{fieldId}`
- The API should not require any authentication
- The API should not do any validation of any sort regarding the field boundary
- The API should not accept any partial updates

Details

- A field entity is modelled like in the below JSON representation:

```
{
    "id": "a0f63e74-d7ef-4924-acb3-0e770ae9ec98",
    "name": "Potato field",
    "created": "2020-07-25T10:03:56.782Z",
    "updated": "",
    "countryCode": "DEU",
    "bounderies": {
        "id": "a0f63e74-d7ef-4924-acb3-0e960ae9ec98",
        "created": "2020-07-25T10:03:56.782Z",
        "updated": "",
        "geoJson": {
            "type": "Feature",
            "properties": {},
            "geometry": {
                "type": "Polygon",
                "coordinates": [
                    [
                        [-5.553604888914691, 33.88229680420605],
                        [-5.5516736984239685, 33.88229680420605],
                        [-5.5516736984239685, 33.88372189858022],
                        [-5.555965232847882, 33.88390003370375],
                        [-5.555965232847882, 33.88229680420605],
                        [-5.553604888914691, 33.88229680420605]
                    ]
                ]
            }
        }
    }
}
```

- If you would like to use another GeoJSON, you can play around using geojson.io by drawing your own field's boundaries, but you can use this example in all the records you make to test your implementation

Retrieve weather history for a given field boundary

As an API consumer, I'd like to be able to retrieve weather history for a given field boundary so that I can assess whether or not my soil needs some action, such as irrigation.

Acceptance criteria

- The API should fetch the weather history for a given polygon from OpenWeather Agro Monitoring
  - https://agromonitoring.com/api/polygons
  - https://agromonitoring.com/api/history-weather
- The API should retrieve temperature, temperature min and max as well as humidity for the last 7 days
- The weather data should be retrieved through a GET request towards `/fields/{fieldId}/weather`
- The performance of the query does not matter for the moment
- The API should not require any authentication

Details

- The response of the API should reflect the JSON below:

```
{
        "weather": [
                {
                        "timestamp": "1485705600",
                        "temperature": 288.15,
                        "humidity": 85,
                        "temperatureMax": 289.16,
                        "temperatureMin": 280.16
                }, {
                        "timestamp": "1485705700",
                        "temperature": 288.15,
                        "humidity": 85,
                        "temperatureMax": 289.16,
                        "temperatureMin": 280.16
                },
                ...
        ]
}
```

Requirements

Should have

- Should be developed using Java 11+ and Springboot
- Should be running within a Docker container
- We should be able to run it locally `docker run ...` or `docker-compose up`
- Should function correctly and honour the contract defined in the above sections
- Should execute the build and the tests successfully
- The delivery must contain the project and a `README.md` file that says at least how to build and run the project locally
- You should add to the `README.md` your notes and assumptions that you made during the development if you think it would help us to see your motivation around technical decisions
- The project should be delivered using a public or provide Git service such as Github, Bitbucket or GitLab in which we should have access to

You choose

- Gradle or Maven, it doesn't matter for us
- The only requirement is to stick with Java 11+ and Spring Boot, so outside of it you may use any library/framework you would like to use to demonstrate your knowledge and experience

Should you have any question

We know that the development lifecycle requires a bunch of back and forth communication among the software engineers and the product owners (in this case, we the recruiters) so should you have any question regarding the requirements please feel free to reach out to daniel.fernandes@yara.com with the subject tag `[Hiring process]: MY SUBJECT GOES HERE`.