Java Developer Challenge

Thank you for your interest in working at car2go! After your first call with us, we'd like to get to know your coding a bit better with this challenge.

We described a small challenge with some acceptance criteria below. Please build a microservice that fulfills these criteria and send us the source code via ZIP, or any private cloud storage.
Don’t publish it on Github or anywhere else.

The goal is to build a microservice which calculates the position of car2go vehicles inside strategic geojson-polygons and serve the cars and polygons via a REST API.

Details:
Build a loop that regularly fetches the car2go vehicles for the location Stuttgart from our API. API is provided in the form of the public Docker image. Which means you need to have Docker installed and operating.

You can have the API up and running using this set of commands (can vary depending on your local configuration):

```
docker pull car2godeveloper/api-for-coding-challenge docker
run -d -p 3000:3000 car2godeveloper/api-for-coding-challenge
```

If the container is running the API swagger specification is available under this url:

http://localhost:3000/documentation/

The vehicles are live data but are not streamed so you should refresh them in an certain interval. Use this json dump for your polygon data for Stuttgart:

https://gist.github.com/codeofsumit/6540cdb245bd14c33b486b7981981b7b

Calculate which cars are in which polygons.
Provide a REST API where polygons and cars can be queried while cars include the polygon ID's they're in, and the polygons include the VIN of cars currently in them.
Use Swagger as the API documentation.
Provide a Dockerfile for docker based deployment.

Please note your decisions, thoughts and comments inside a README.md.

Some pointers to help you along the way:

- We will look at how you used framework specific techniques
- We will look at clean implementations, performance and maintainable architecture
- We care for current technologies
- Please include a local git commit history.
- Write tests where you feel they make sense. Focus on Unit Tests for your business logic.
- Put decisions regarding dependencies and architecture in the README.md. Be open about issues with the API, your application or general thoughts.

- Write production-ready code. Pretend like you would ship this code to real customers and might have to maintain it in the future.

We hope you have a lot of fun during this challenge and getting a taste of how it feels to work with real cars on the roads!
We're looking forward to see your code.

Good luck
- The car2go team