Lab 6. Feign and Registry

Part 1: Feign

Write 2 services:

1. A StockService that returns how many products we have in stock given a certain productNumber. The number that is returned can be hard coded. This service runs on port 8900.
2. A ProductService that returns a Product given a certain productNumber. The Product contains the fields: productNumber, name, number on stock
   The number on stock is retrieved from the StockService. This service runs on port 8901.

Use Feign so that the ProductService can call the StockService.

The ProductService needs the Feign dependency:

```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-openfeign</artifactId>
</dependency>
```

Part 2

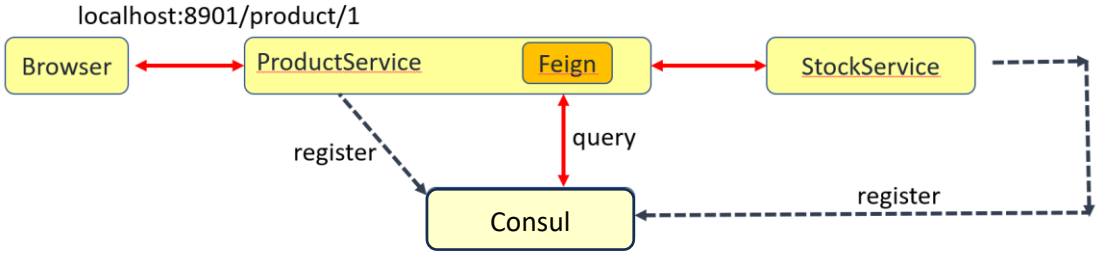
Then configure these The ProductService and the StockService so that they use the Consul registry. Both services need the following dependencies:

```
<dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-web</artifactId>
</dependency>
<dependency>
   <groupId>org.springframework.cloud</groupId>
   <artifactId>spring-cloud-starter-consul-discovery</artifactId>
</dependency>
<dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
```

First run the Consul registry with the docker command:
docker run -d -p 8500:8500 --name=consul hashicorp/consul:latest

Run the StockService and see in the Consul dashboard (http://localhost:8500/) if it is registered.
Run the ProductService and see in the Consul dashboard if it is registered.

Then if you call the ProductService, it should call to the StockService using the registry.



Part 3

In Lab 4 part 2 we implemented a modular monolith with a Product component and a shoppingcart component. Take the solution of lab 4 part 2 and modify it to a microservice architecture. This means we get a separate project for the product service and a separate project for the shoppingcart service. Then we can use the client project that call these separate components.

Make sure that when we add a product to the shopping service, that the shopping service calls the product service to get product data. (Make sure the components use the registry to call each other)

Also the client will update the price of a product in the product service. Make sure that the corresponding products in the shoppingcarts in the shoppingservice will also be updated.

What to hand in?

1. A zip file containing all service projects for part 1
2. A zip file containing all service projects for part 2
3. A zip file containing all service projects for part 3