Tasks and Assignments Course 02267

Hubert Baumeister

January 2024

1 Task (group): Simple DTU Pay with a REST API TASK_REST2

The task is to implement a REST backend of a simplified payment system called "Simple DTU Pay". It is possible to create payments between customers and merchants by providing their respective id's and an amount. Furthermore, it should be possible to ask the system for a list of payments.

A possible way to express the payment scenario as a Cucumber scenario is:

```
Feature: Payment
Scenario: Successful Payment
  Given a customer with id "cid1"
  And a merchant with id "mid1"
  When the merchant initiates a payment for 10 kr by the customer
  Then the payment is successful
```

Here is an example of the associated step definitions.

```
public class SimpleDTUPaySteps {
   String cid, mid;
   SimpleDTUPay dtuPay = new SimpleDTUPay();
   boolean successful;

   @Given("a customer with id {string}")
   public void aCustomerWithId(String cid) {
      this.cid = cid;
   }

   @Given("a merchant with id {string}")
   public void aMerchantWithId(String mid) {
      this.mid = mid;
   }

   @When("the merchant initiates a payment for {int} kr by the customer")
   public void theMerchantInitiatesAPaymentForKrByTheCustomer(int amount) {
      successful = dtuPay.pay(amount,cid,mid);
   }

   @Then("the payment is successful")
   public void thePaymentIsSuccessful() {
      assertTrue(successful);;
   }
}
```

Note that the `pay` method in class `SimpleDTUPay` uses, e.g., the JAX-RS framework to call the SimpleDTUPay REST API on the server. Hiding the REST calls in their own class helps to keep the step definitions simple and also avoids repetitions, if the same method is called several times in the tests.

```
Scenario: List of payments
  Given a successful payment of "10" kr from customer "cid1" to merchant "mid1"
  When the manager asks for a list of payments
  Then the list contains a payments where customer "cid1" paid "10" kr to merchant "mid1"
```

The application knows a customer with id `cid1` and a merchant with id `mid1` (no other informations of a customer and a merchant are relevant at this moment). No other customers and merachants are known. Using other customers and merchants than `cid` and `mid` should result in a failure.

```
Scenario: Customer is not known
  Given a customer with id "cid2"
  And a merchant with id "mid1"
  When the merchant initiates a payment for "10" kr by the customer
  Then the payment is not successful
  And an error message is returned saying "customer with id cid2 is unknown"
```

There should be a similar scenario for an unknown merchant.

Your task is to design the proper REST interface. What are the resources, e.g. payments, ...? How can these resources be manipulated, e.g. creating a new payment, listing all payments, and how are those realized using HTTP Verbs.

You should define the needed resources, their URI (note these should not contain any verbs, e.g. `pay`), the HTTP verb and which function they trigger.

Have a look at the design of the REST interface for the StudentRegistration as an example.

1.1 Tips

No database is required. You can store payments in a list in a field.

You can build Simple DTU Pay based on a copy of the code from the previous task. Make sure though that you rename the projects accordingly in the `pom.xml` files and remove parts that are not used anymore.

It is important to only change small things and make sure that the resulting system is working. You notice that your steps are too big, when your application stops working and it is not immediatly clear what caused the application to malfunction. In this case, go back to the last working version and use smaller steps.