

DELFT UNIVERSITY OF TECHNOLOGY
GROUP 51

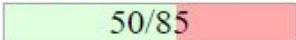
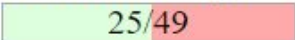
Assignment 3

22.01.2021

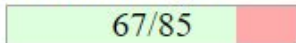
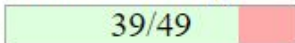
1) Increasing the Mutation Coverage for ProductController class

(Path: *transactions/src/main/java/nl.tudelft.sem.transactions/controllers/ProductController*)

We have identified the ProductController class as one of the four classes with Mutation Coverage less than 70% in our project. Before, it had a score of 51% (screenshot from the PIT report below) and we wanted to increase it by at least 20%.

Name	Line Coverage	Mutation Coverage
<u>ProductController.java</u>	59% 	51% 

We have managed to do so by fixing the existing test cases and adding 13 new test cases. As a result, our Mutation Coverage has risen to 80% meaning an improvement of 29% (screenshot from the PIT report below).

Name	Line Coverage	Mutation Coverage
<u>ProductController.java</u>	79% 	80% 

We haven't found any equivalent mutants in this class.

[Link to the corresponding commit](#)

2) Increasing the Mutation Coverage of TransactionController class:

(Path: transactions/src/main/java/nl.tudelft.sem.transactions/controllers/TransactionController)

One of the classes with the lowest Mutation coverage was the Transaction Controller, with 33%. However, as this class is in the “transaction” microservice and it is responsible for getting food out of the fridge of the house, it is very important and its functionalities are used a lot. This is the reason why we decided to increase the Mutation coverage of this class. One of the main problems with the increase of the mutation coverage in this class was that we have to mock the class responsible for the communication between the “transaction” and “request” microservices. Therefore, we had to mock the server in order to establish this communication.

As you can see on the images below, with research and a lot of improvements, during the week, we managed to improve our Mutation Coverage score a couple of times and we finally reached 100% Mutation Coverage. This result gave us a lot of confidence that one of the main functionalities in our application works as expected and that there are not any bugs and issues in the TransactionController. In the images below you can see our process of improvement during the week.

[Link to the corresponding commit](#)

Pit Test Coverage Report

Package Summary

nl.tudelft.sem.transactions.controllers

Number of Classes	Line Coverage	Mutation Coverage
2	65% <div><div></div><div></div></div> 87/134	44% <div><div></div><div></div></div> 35/79

Breakdown by Class

Name	Line Coverage	Mutation Coverage
ProductController.java	59% <div><div></div><div></div></div> 50/85	51% <div><div></div><div></div></div> 25/49
TransactionController.java	76% <div><div></div><div></div></div> 37/49	33% <div><div></div><div></div></div> 10/30

Report generated by [PIT](#) 1.5.1



Pit Test Coverage Report

Package Summary

nl.tudelft.sem.transactions.controllers

Number of Classes	Line Coverage	Mutation Coverage
2	66% <div><div></div><div></div></div> 88/134	51% <div><div></div><div></div></div> 40/79

Breakdown by Class

Name	Line Coverage	Mutation Coverage
ProductController.java	59% <div><div></div><div></div></div> 50/85	51% <div><div></div><div></div></div> 25/49
TransactionController.java	78% <div><div></div><div></div></div> 38/49	50% <div><div></div><div></div></div> 15/30

Report generated by [PIT](#) 1.5.1



Pit Test Coverage Report

Package Summary

nl.tudelft.sem.transactions.controllers

Number of Classes	Line Coverage	Mutation Coverage
2	66% <div><div></div><div></div></div> 89/134	58% <div><div></div><div></div></div> 46/79

Breakdown by Class

Name	Line Coverage	Mutation Coverage
ProductController.java	59% <div><div></div><div></div></div> 50/85	51% <div><div></div><div></div></div> 25/49
TransactionController.java	80% <div><div></div><div></div></div> 39/49	70% <div><div></div><div></div></div> 21/30

Report generated by [PIT](#) 1.5.2



Pit Test Coverage Report

Package Summary

nl.tudelft.sem.transactions.controllers

Number of Classes	Line Coverage	Mutation Coverage
2	72% 96/133	69% 54/78

Breakdown by Class

Name	Line Coverage	Mutation Coverage
ProductController.java	59% 50/85	51% 25/49
TransactionController.java	96% 46/48	100% 29/29

Report generated by [PIT](#) 1.5.2

3) Increasing the Mutation Coverage of TransactionsSplitCredits class:

(Path: *transactions/src/main/java/nl.tudelft.sem.transactions/entities/TransactionsSplitCredits*)

From the images below you can see that in the “transactions” microservice, the TransactionsSplitCredits entity is the only one that does not have 100% Mutation Coverage. This inspired us to improve the testing of this class.

The main problem in this class was that it does not have coverage on some of the methods. Therefore, the solution to this problem was to add four more tests. By adding them we managed to check all of the return statements of the methods in the class and this way we killed all of the mutants. As you can see on the last image, by adding those tests we managed to achieve 100% Mutation coverage on the TransactionsSplitCredits controller, as well as 100% Mutation coverage on the whole entities package in the “transaction” microservice.

Pit Test Coverage Report

Package Summary

nl.tudelft.sem.transactions.entities

Number of Classes	Line Coverage	Mutation Coverage
3	97% <div><div></div><div></div></div> 66/68	79% <div><div></div><div></div></div> 15/19

Breakdown by Class

Name	Line Coverage	Mutation Coverage
Product.java	100% <div><div></div><div></div></div> 37/37	100% <div><div></div><div></div></div> 8/8
Transactions.java	88% <div><div></div><div></div></div> 15/17	100% <div><div></div><div></div></div> 5/5
TransactionsSplitCredits.java	100% <div><div></div><div></div></div> 14/14	33% <div><div></div><div></div></div> 2/6

Report generated by [PIT](#) 1.5.1



Pit Test Coverage Report

Package Summary

nl.tudelft.sem.transactions.entities

Number of Classes	Line Coverage	Mutation Coverage
3	100% <div><div></div><div></div></div> 68/68	100% <div><div></div><div></div></div> 19/19

Breakdown by Class

Name	Line Coverage	Mutation Coverage
Product.java	100% <div><div></div><div></div></div> 37/37	100% <div><div></div><div></div></div> 8/8
Transactions.java	100% <div><div></div><div></div></div> 17/17	100% <div><div></div><div></div></div> 5/5
TransactionsSplitCredits.java	100% <div><div></div><div></div></div> 14/14	100% <div><div></div><div></div></div> 6/6

Report generated by [PIT](#) 1.5.2

[Link to the corresponding commit](#)

4) Increasing the Mutation Coverage for HouseController class

(Path: requests/src/main/java/nl.tudelft.sem.requests/controllers/HouseController)

As it can be observed in the images below, we have found that the HouseController class from the Requests microservice had a mutation score lower than 70%. We have succeeded in doing so and achieving a Mutation coverage of 88%.

The main problem we have found in this class was that the cases where exceptions were thrown were not thoroughly tested and that there was a method, getHouseByUsername, which was not at all tested. We have written 6 new tests and applied some changes to the tests that were already written in order to improve the Mutation coverage. As can be observed from the second image, we have managed to improve this score by 23%.

Pit Test Coverage Report

Package Summary

nl.tudelft.sem.requests.controllers

Number of Classes	Line Coverage	Mutation Coverage
3	87% 155/179	67% 66/98

Breakdown by Class

Name	Line Coverage	Mutation Coverage
HouseController.java	80% 76/95	65% 34/52
RequestController.java	100% 21/21	75% 12/16
UserController.java	91% 52/57	67% 20/30

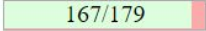
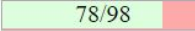
Report generated by [PIT](#) 1.5.1



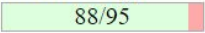
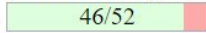

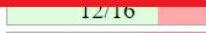
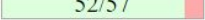
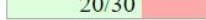
Pit Test Coverage Report

Package Summary

nl.tudelft.sem.requests.controllers

Number of Classes	Line Coverage	Mutation Coverage
3	93% 	80% 

Breakdown by Class

Name	Line Coverage	Mutation Coverage
HouseController.java	93% 	88% 
RequestController.java	100% 	75% 
UserController.java	91% 	67% 

Report generated by [PIT](#) 1.5.1

[Link to the corresponding commit](#)