Delft University of Technology

SUBJECT: Software Engineering Methods
COURSE CODE: CSE2115

Assignment 3

Authors:
Ben Carp
Denis Corlade
Andrejs Karklins
Daniel Poolman
Justin Rademaker
Auke Sonneveld

OP15-SEM29
January 22, 2021

1 Introduction

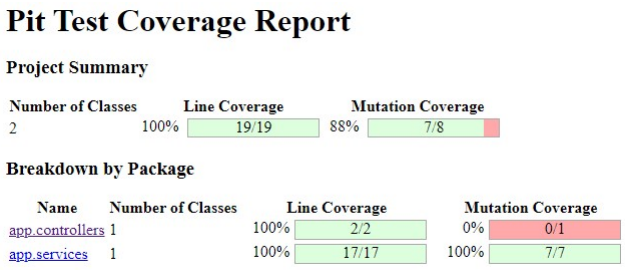
Mutation testing is used to design new software tests and evaluate the quality of existing software tests. It involves modifying a program in small ways. These mutations are automatically seeded into your code, and then the tests are run. If the test fails, this means that it caught the mutation. We then say that 'the mutant has been killed'. If the test passes, this is a sign that the test suit should be improved. An exception to this rule is in the case of 'equivalent mutants' which provide the same functionality despite the changes.

Mutation testing adds extra value to the test suite. Traditional test coverage over lines, statements and branches measures which code is executed by your tests but it does not check whether your tests are actually able to detect fault in the executed code. Mutation testing helps detect whether the code has been meaningfully tested.

For this assignment, we have chosen Pitest as our mutation testing tool as it has the ability to generate readable HTML reports, while also having an IntelliJ plugin available. We found Pitest to be the leading mutation tool compared to other mutation testing systems for Java. It is extremely easy to use and it is fast. Besides all of that, it also makes use of different illustrations to easily identify missing mutants or lines that are already covered.

2 Already good

Most of the classes already have a good mutation score, however as you can see on the figures 1-5, some of the packages have very low coverage. This is because some classes contain configuration files, or external library calls, which cannot be tested properly. Therefore, we decided to not add these 5 packages to the list of things we could improve.



Pit Test Coverage Report

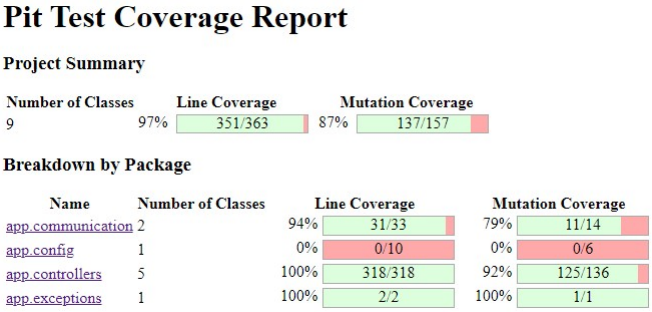
Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
2	100%	19/19	88%	7/8

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
app.controllers	1	100%	2/2	0%	0/1
app.services	1	100%	17/17	100%	7/7

Figure 1: Authorization micro-service



Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
9	97%	351/363	87%	137/157

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
app.communication	2	94%	31/33	79%	11/14
app.config	1	0%	0/10	0%	0/6
app.controllers	5	100%	318/318	92%	125/136
app.exceptions	1	100%	2/2	100%	1/1

Figure 2: Exam micro-service

Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
6	99%	91/92	100%	47/47

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
app.communication	1	96%	23/24	100%	8/8
app.controllers	5	100%	68/68	100%	39/39

Figure 3: Student micro-service

Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
6	99%	140/141	98%	78/80

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
app.communication	1	96%	23/24	75%	6/8
app.controllers	5	100%	117/117	100%	72/72

Figure 4: Teacher micro-service

Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
14	97%	479/493	98%	194/198

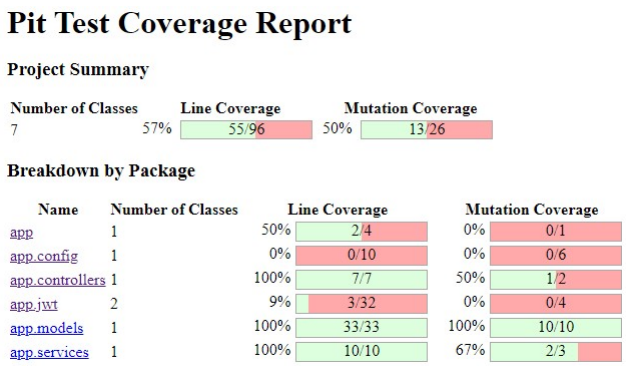
Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
app.json	4	98%	331/339	98%	184/188
app.models	10	96%	148/154	100%	10/10

Figure 5: Shared module

3 Improvements

Now we focus on Authentication and Course micro-services, they did have problems and have been fixed. In the figures 6-7 you can see their scores before changes to the tests.



Pit Test Coverage Report

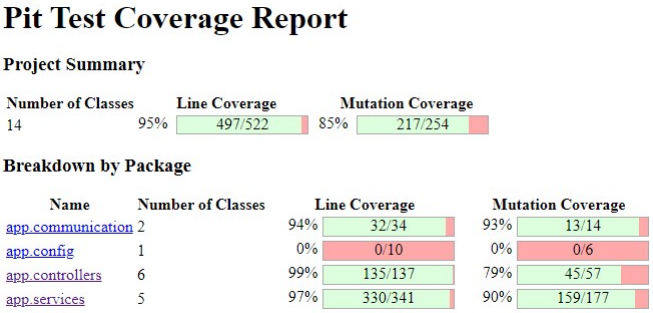
Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
7	57%	55/96	50%	13/26

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
app	1	50%	2/4	0%	0/1
app.config	1	0%	0/10	0%	0/6
app.controllers	1	100%	7/7	50%	1/2
app.jwt	2	9%	3/32	0%	0/4
app.models	1	100%	33/33	100%	10/10
app.services	1	100%	10/10	67%	2/3

Figure 6: Authentication micro-service



Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
14	95%	497/522	85%	217/254

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
app.communication	2	94%	32/34	93%	13/14
app.config	1	0%	0/10	0%	0/6
app.controllers	6	99%	135/137	79%	45/57
app.services	5	97%	330/341	90%	159/177

Figure 7: Course micro-service

3.1 Course service

In course service there were 3 controllers that stood out to us, because they had tests which did not assert the result correctly. Mutants that survived were not tested for nulls that could get returned in process, so we have written new tests for the following controllers. These assert the result correctly in the new situation:

- AnswerController (50% -> 100%)
- QuestionController (20% -> 100%)
- TopicController (25% -> 100%)

The final commit can be found with this link: Final commit

3.2 Authentication service

The UserDetailsServiceImpl method was not properly tested, which could lead to a future fault in the system. The mutant survived due to an untested if-statement. The inversion of the boolean comparator did not lead to failure of any test. This mistake has been fixed by adding one more test. In the new situation, the coverage is 3/3 in authentication services, which is an improvement of 33%.

- UsersDetailsServiceImpl (67% -> 100%)

The final commit can be found with this link: Final commit

4 Images after refactoring

Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
7	57%	55/96	54%	14/26

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
app	1	50%	2/4	0%	0/1
app.config	1	0%	0/10	0%	0/6
app.controllers	1	100%	7/7	50%	1/2
app.jwt	2	9%	3/32	0%	0/4
app.models	1	100%	33/33	100%	10/10
app.services	1	100%	10/10	100%	3/3

Figure 8: Authentication micro-service after

Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
14	95%	501/526	83%	212/254

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
app.communication	2	94%	32/34	93%	13/14
app.config	1	0%	0/10	0%	0/6
app.controllers	6	99%	139/141	79%	45/57
app.services	5	97%	330/341	87%	154/177

Figure 9: Course micro-service after