DP2 2023-2024
Analysis report D04

Acme Software Factory

Repository: https://github.com/rafcasceb/Acme-SF-D04

Student #4:

- Mellado Díaz, Luis luimeldia@alum.us.es

Other members:

- Flores de Francisco, Daniel danflode@alum.us.es
- Heras Pérez, Raúl rauherper@alum.us.es
- Vento Conesa, Adriana adrvencon@alum.us.es
- Castillo Cebolla, Rafael rafcasceb@alum.us.es

GROUP C1.049

Version 1.0

25-05-24

Content Table

Executive summary

This report will provide an in-depth examination of both the testing process and its outcomes. It will contain sections dedicated to functional testing and performance testing.

We will follow a precise but straightforward approach aiming to promote comprehension and ensure the quality of the product.

Revision Table

Date	Version	Description of the changes	Sprint
12/05/2024	1.0	• Executive summary • Introduction • Functional testing	4
16/05/2024	1.0	• Performance testing • Conclusion • Bibliography	4
25/05/2024	1.1	• Revision after upgrading the framework to 24.5.0	4

Introduction

This document will provide a detailed analysis of the testing procedure and results for the following features:

- Operations by sponsors on Sponsorships.
- Operations by sponsors on Invoices.

The content of a testing report is organized into two chapters:

- Functional Testing: This section consists in a compilation of implemented test cases categorized by feature. Each test case is accompanied by a brief description and an assessment of its effectiveness in bug detection.

- Performance Testing: This segment includes informative charts and a 95% confidence interval for the project's wall time in serving requests during functional tests. Additionally, a 95% confidence hypothesis contrast is provided.

Contents

Functional testing

Operations by sponsor on Sponsorships

Test case 1: list-mine

For the list command, we just performed the listing of sponsorships for the two existing users with the sponsor role.

For hacking, we considered accessing the URL by a wrong role. It is impossible to test the correct role wrong user situation for this command since it is a “mine” type.

It provided a coverage of 93.5%, covering all instructions except a default assertion, which is logical. No errors were detected.

Test case 2: show

Test case number 2 consisted of showing sponsorships belonging to sponsor1.

For hacking, both an anonymous user and sponsor 2 tried to access, edit, delete an publish a sponsor1 sponsorship.

It provided a coverage of 97%, covering all instructions except a default assertion, which is logical. No bugs were detected.

Test case 3: create

For this command, we tried to create a sponsorship with all type from invalid data. Starting from the blank form each attribute has been put to the limits. Once we checked the system rejects invalid data, we provided valid inputs.

It provided a coverage of 96%, covering all instructions except a default assertion, which is logical. No bugs were detected.

Test case 4: update

For this command, we have updated the sponsorship AAC-723. For each attribute we have checked the system rejects all different types of invalid data. Later, for each attribute, we have checked the system accepts all different types of valid data.

It provided a coverage of 93.4%. The default assertions and the validation that checks that a published sponsorship cannot be updated were not covered (this makes sense, since a well-intentioned user cannot and would not perform that action from the user interface).

A small bug was detected related to the duplicated code if the field was left in blank.

Test case 5: delete

For delete command we deleted a couple pages of sponsorships and the special case of trying to delete a sponsorship with a published invoice was checked.

It provided a coverage of 92.9 %. It covered all instructions but some logical exceptions: default assertions and the validation that checks that it is not possible to delete a published sponsorship. That validation was completely useless since that case was already checked in the authorization process. It was deleted.

Test case 6: publish

The procedure to test the publish command was the following:

1. Proceed in the same way as we did for the update command. Test that the system rejects invalid data and accepts valid inputs.
2. We put to the test the business rule that states that a sponsorship cannot be published I the invoices do not sum the amount.
3. We published one of the invoices associated with the sponsorship with a specific currency (EUR). Then we tried to publish the sponsorship in a different currency (USD).

It provided a coverage of 96.4%. It covered all instructions except for the default not null assertion. The validation that checks that you cannot publish an already published sponsorship is not covered for the same reasons as before.

Bugs were detected when trying to compute the total sum of the invoices if the quantity of the invoice was an invalid value.

Image 1: Sponsorship feature coverage

Operations by sponsor on Invoices

Test case 1: list-mine

All the invoices for the two users with sponsor role were listed.

For hacking we considered accessing the lists with a different role. Trying to access an invoice list with a different sponsor is not possible since it is a list-mine.

It provided a coverage of 93.3 %, covering all instructions except a default assertion, which is logical. No bugs were detected.

Test case 2: list-for-sponsorship

The button to see the associated invoice was pressed for different sponsorships, including sponsorships with 0, 1 or several invoices associated.

For hacking, we tried accessing the same user story listing with a wrong role and with sponsor 2.

It provided a coverage of 92.6 %, covering all instructions except a default assertion, which is logical. No bugs were detected.

Test case 3: show

In this test case we showed several pages of invoices.

For hacking, both an anonymous user and sponsor 2 tried to access, edit, delete and publish a sponsor1 invoice.

It provided a coverage of 94.9 %, covering all instructions but some logical except the default assertions. No bugs were detected.

Test case 4: create

For the create command, we tried to create an invoice starting from a blank form and checking that the system rejects invalid data. Next, we checked all ranges of valid inputs.

It provided a coverage of 95.1 %, covering all instructions except a default assertion, which is logical. No bugs were detected.

Test case 5: update

In this test case we tried to update an invoice, checking the rejection of invalid inputs and putting the ranges of valid data to the limits.

It provided a coverage of 94.9 %. It covered all instructions except for the default not null assertions and the validation that checks that a published invoice cannot be published. This happens because it is impossible that a well-intentioned user can perform that action using the user interface.

After realizing that the mentioned validation is unnecessary (since it is already checked in the authorization), it was deleted and the coverage went up to 95.5%. There is still a line not covered in the unbind, this code was added at the beginning of the development to fix an error that I haven't been able to reproduce anymore. I did not remove it just in case.

Test case 6: delete

A couple pages of invoices were deleted.

It provided a coverage of 66.43 %. Again, the validation that checks that a published invoice cannot be deleted never occurs. On top of that, since there are no restrictions for the deletion of an invoice, the unbind is not covered.

Both the unbind and the published restrictions are unnecessary and were deleted. This improvement provided a coverage of more than 89.5%.

Test case 7: publish

In this test case we proceed in the following way:

1. Test the publish command as if it were the update command. Check the invalid inputs are rejected and that valid data is accepted.
2. There is a special condition to be checked. The currency of an invoice must match the currency of the sponsorship it is associated with.

It provided a coverage of 93.6%. The validation that checks that a published invoice cannot be published again never occurs. After deleting it we achieved a coverage of 95.5%. There is still a line not covered in the unbind, this code was added at the beginning of the development to fix an error that I haven't been able to reproduce anymore. I did not remove it just in case.

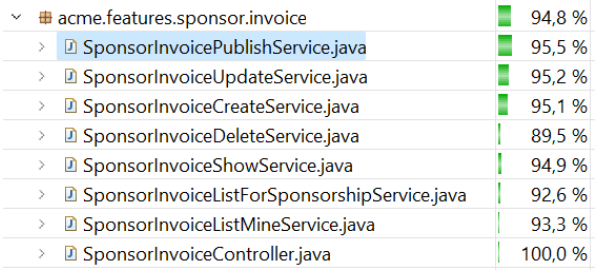

Image 2: Invoice feature coverage.

Performance testing

Let's start by reviewing the performance data gathered during functional testing. Initially, we'll outline the outcomes of the tests, followed by an examination of a hypothesis contrast.

Performance data

These tests have been recorded in a computer with the following characteristics: 11th Gen Intel® Core™ i5-1135G7 CPU @2.60 GHz and 8 GB of RAM.

Following the completion of the tests, a graph has been produced illustrating the mean response time for each request pathway.

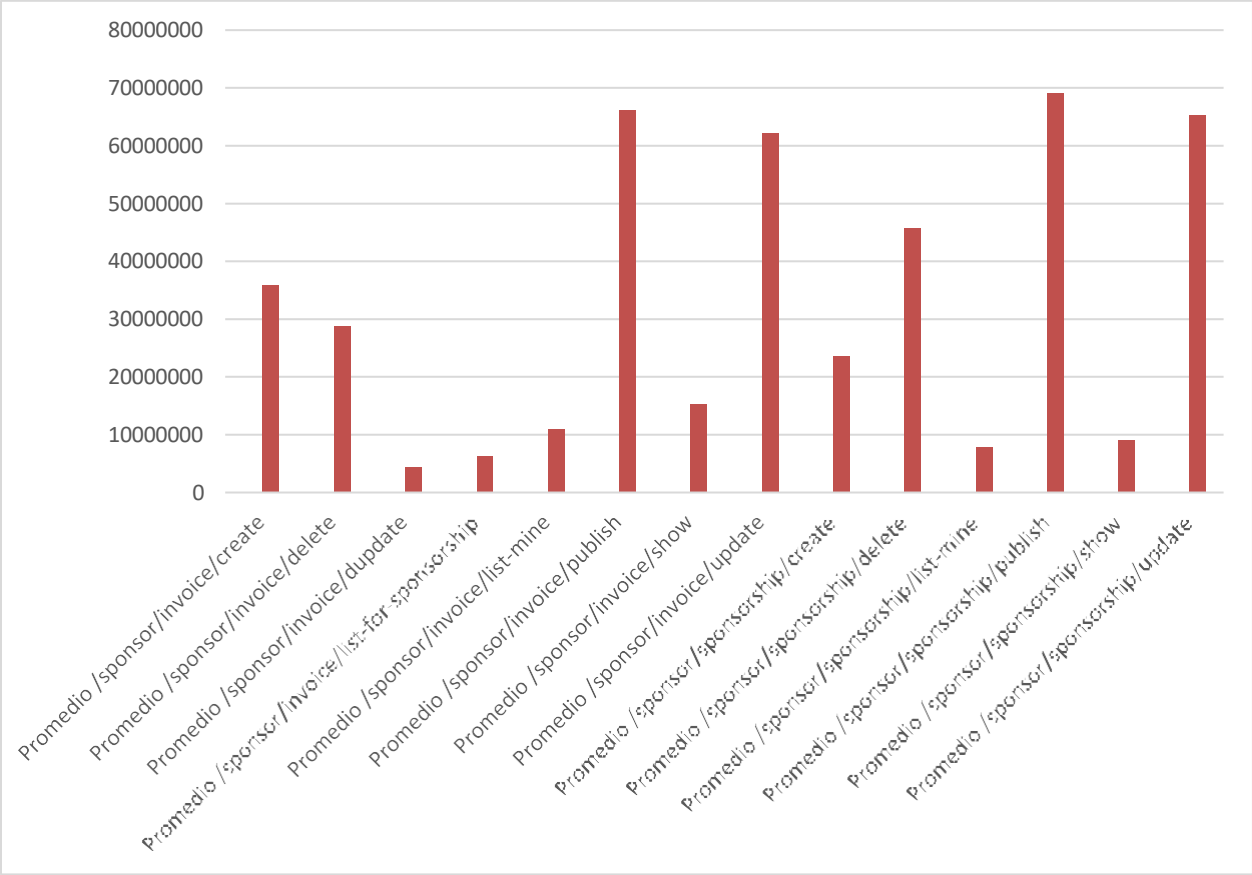

Image 3: Average response time per request path.

What we observe is that the requests that take the most time are the publish and update requests. This makes sense since they contain more complicated validations, specially the publish requests that. For example, the sponsorship publish request, checks if the sum of all published invoices amount sums up to the sponsorship price.

The deletion of sponsorship is also a curious case. It takes longer since it is not possible to erase a sponsorship.

The rest of display features show a consistent and logical performance. Let us now calculate a confidence interval for the whole test suite.

	A	B	C	D	E	F
1	*Columna1*					
2				Interval (ms)	30,6206068	26,4826795
3	Media	28,55164313		Interval (s)	0,03062061	0,02648268
4	Error típico	1,053999901				
5	Mediana	11,131				
6	Moda	1,6071				
7	Desviación estándar	29,68090671				
8	Varianza de la muestra	880,9562233				
9	Curtosis	3,041007833				
10	Coeficiente de asimetría	1,313872098				
11	Rango	239,8146				
12	Mínimo	1,4257				
13	Máximo	241,2403				
14	Suma	22641,453				
15	Cuenta	793				
16	Nivel de confianza(95,0%)	2,068963632				

Image 4: Data analysis summary and confidence level.

Based on our data analysis summary, we calculated the 95.0% confidence interval for our data, which ranges from 26.48 to 30.62 milliseconds.

Although this project lacks a specific performance benchmark to compare this confidence interval against, this response time is generally deemed acceptable.

Hypothesis contrast

Since there are no specific performance requirements for this delivery, we will simulate a hypothesis test. We will generate new data by increasing the actual test data by 10%.

First, we will create a data analysis summary for both performance samples and then compare the results.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	*BEFORE*								*AFTER*					
2				Interval (ms)	30,6206068	26,4826795						Interval (ms)	33,6826674	29,1309474
3	Media	28,55164313		Interval (s)	0,03062061	0,02648268			Media	31,4068074		Interval (s)	0,03368267	0,0291309
4	Error típico	1,053999901							Error típico	1,15939989				
5	Mediana	11,131							Mediana	12,2441				
6	Moda	1,6071							Moda	1,76781				
7	Desviación estándar	29,68090671							Desviación estándar	32,6489974				
8	Varianza de la muestra	880,9562233							Varianza de la muestra	1065,95703				
9	Curtosis	3,041007833							Curtosis	3,04100783				
10	Coeficiente de asimetría	1,313872098							Coeficiente de asimetría	1,3138721				
11	Rango	239,8146							Rango	263,79606				
12	Mínimo	1,4257							Mínimo	1,56827				
13	Máximo	241,2403							Máximo	265,36433				
14	Suma	22641,453							Suma	24905,5983				
15	Cuenta	793							Cuenta	793				
16	Nivel de confianza(95,0%)	2,068963632							Nivel de confianza(95,0%)	2,27586				

Image 5: Data analysis summary and confidence level for both samples.

The results have naturally increased, which might be viewed as a decline in performance. However, intuitively comparing the confidence intervals is challenging. Therefore, we will use a Z-Test to facilitate this comparison. Here are the results:

	A	B	C
1	Prueba z para medias de dos muestras		
2			
3		*Time (miliseconds)*	*Time (miliseconds) 10%*
4	Media	28,55164313	31,40680744
5	Varianza (conocida)	880,9562233	1065,95703
6	Observaciones	793	793
7	Diferencia hipotética de las medias	0	
8	z	-1,822193107	
9	P(Z<=z) una cola	0,034212845	
10	Valor crítico de z (una cola)	1,644853627	
11	Valor crítico de z (dos colas)	0,06842569	
12	Valor crítico de z (dos colas)	1,959963985	

Image 6: Z-Test results between the two performance samples.

A Z-Test relies on a value called alpha, which is calculated as 1 minus the confidence level percentage. For this case, alpha will be 0.05. To interpret the Z-Test result, we compare the two-tail p-value (Valor crítico de z (dos colas) in Spanish) to alpha. If the p-value is below alpha, we then compare the averages to determine if the new average has decreased.

In our case, however, the p-value is 0.06842569, which is higher than alpha. This indicates that the changes we implemented did not lead to any significant improvement. Although the sample times differ, they are generally considered equivalent overall.

Conclusions

This rigorous formal testing process enabled us to examine our code to uncover any bugs. We identified two minor mistakes in validation methods that had to do with comparing null dates. Overall, the test outcomes were highly favorable, achieving nearly complete instruction coverage, except for some logical branches, and demonstrating excellent performance, including a remarkable average response time.

Bibliography

Intentionally blank.