26 DE MAYO DE 2024

TESTING REPORT STUDENT #4

GROUP C-1.047

Acme SF, Inc.

Repository link: https://github.com/Cargarmar18/Acme-SF

Manuel Castillejo Vela
Email: mancasvel@alum.us.es

Table of contents

1. Executive summary

The report encompasses two critical aspects in our testing process: functional testing and performance testing. Each chapter provides a thorough view of the respective tests, along with their findings and implications for the application.

The Functional Testing chapter documents the tests executed for each feature, including comprehensive descriptions of their functions. These tests were mainly effective, revealing predominantly minor bugs. Correcting these issues substantially improved the application's overall performance.

Conversely, the Performance Testing chapter offers a detailed evaluation of the time the project takes to handle requests on a computer, nothing that one set of requests is 10% slower than the other. This chapter features detailed charts and 95% confidence intervals, which are indispensable for optimizing and assessing the system's performance.

2. Revision table

Revision number	Date	Description
1.0.0	20-05-2024	Creation of the document.

3. Introduction

This document thoroughly analyzes the testing and performance evaluation of the mandatory features developed by Student #4. The goal is to ensure that the functionalities work correctly, are secure, and operate efficiently by closely examining each test case and performance metric.

We start with Functional Testing, which covers two main areas. The first area, Test Case Analysis, looks at the specific test cases for different features like project and user story management. Each test case is described in detail, explaining the scenarios tested, the methods used, and how effective they were in finding bugs. This includes both regular and edge cases, as well as security measures to ensure only authorized roles can perform certain actions.

Then, the Coverage Analysis gives a quantitative look at how thoroughly the tests cover different classes. A detailed table shows the coverage percentages for various services related to project and user story management, highlighting the comprehensiveness of the testing process.

Next, we move on to Performance Testing, which examines how efficiently the system handles various actions by sponsors. This section identifies the most time-consuming actions and explains why some operations might be less efficient.

As part of this performance evaluation, the Confidence Interval subsection provides the calculations for the 95% confidence interval for the time it takes the system to serve requests. This helps in understanding how consistently the system performs.

Finally, the Hypothesis Contrast subsection explores a hypothetical scenario where system efficiency improves by 10%. Using Z-testing analysis, it assesses whether the observed performance changes are statistically significant, providing a data-driven approach to understanding potential performance improvements.

4. Contents

4.1Testing analysis:

In this section, we will analyze the test suite generated for the mandatory features for Student #4 of testing. This analysis is divided into two parts: the first part focuses on examining each individual test case, while the second part addresses the coverage analysis.

4.1.1 Test case analysis:

Create Sponsorship:

To verify the functionality of creating a sponsorship, I conducted tests by completing the form with various inputs in the capacity of the sponsor role. This included submitting an empty form, entering incorrect values, and testing custom constraints to ensure the system's robustness highlighting the limits. Additionally, we examined how the feature handled edge cases, such as the earliest and latest possible dates, as well as potential SQL injection attempts.

List Sponsorship:

The process of accessing the sponsorship listings according to the the sponsor that created them was tested to ensure the feature's correct implementation. Interaction with the listings page of the sponsor's projects was carried out. Furthermore, security was assessed

by attempting to access the listings without a sponsor role, which resulted in the expected error message indicating unauthorized access for other roles.

Show Sponsorship:

To confirm the accuracy of the feature for viewing specific sponsorship details, we accessed data from multiple sponsorships belonging to a sponsor. Security measures were tested by attempting to access this data without a sponsor role, using the credentials of another sponsor (correct role, incorrect user), and querying an ID not corresponding to any project. Expected errors were received, effectively blocking unauthorized access.

Update Sponsorship:

The update functionality for a sponsorship was verified by submitting various form combinations as a sponsor. This included testing empty forms, incorrect values, and respecting custom constraints. Special attention was given to a custom constraint that retains pre-update codes marked as "taken" in the database. Edge cases, such as the first and last possible values and SQL injection attempts, were also tested. The framework's limitations precluded further POST hacking tests.

Delete Sponsorship:

The deletion of a specific sponsorship, while in draft mode, was tested to verify the feature's proper implementation. Interaction with the specific sponsorship and subsequent deletion confirmed that the function operates correctly. Also we tested that we cannot delete a project that had been already published.

Publish Sponsorship:

The functionality for publishing a sponsorship was tested by attempting to publish projects under various constraints. The main ones were that a sponsorship could not be published with no invoices, without published invoices and when the sum of the total value of the published invoices is not equal to the actual value of the sponsorships. Also some post hackings proves have been made in order to test that we cannot publish an already publish sponsorship

Create Invoice:

To verify the creation of an invoice as a sponsor, we tested the form with multiple input combinations. This included submitting empty forms, incorrect values, and ensuring adherence to custom constraints. The feature's handling of edge cases, such as the first and last possible dates, and potential SQL injection attempts, were also examined.

List Invoices:

The process of accessing invoice listings for user stories was tested to ensure proper implementation. Interaction with the listings page of the sponsor’s invoices related to a sponsorship were carried out. For security, attempts to access the listings of the according

invoices without a sponsor role resulted in the expected error message indicating unauthorized access for other roles or other sponsor without permission.

Show Invoices:

Viewing specific invoice details for user stories was tested by accessing data from multiple invoices belonging to a sponsor. Security assessments included attempts to view this data without a sponsor role, using another sponsor's credentials (correct role, incorrect user), and querying an ID not corresponding to any invoice respectively. Expected errors were received, effectively blocking unauthorized access.

Update Invoice:

The update functionality for an invoice was verified by submitting various form combinations as a sponsor. This included testing empty forms, incorrect values, and ensuring adherence to custom constraints. Edge cases, such as the first and last possible values and SQL injection attempts, were also tested to ensure proper handling. We also tried to update an invoice that was already published obtaining that the action was not allowed.

Delete Invoice:

To verify the deletion of an invoice, interaction with a specific invoice was conducted, and deletion was attempted while it was in draft mode. This confirmed the feature's correct operation. Additional POST hacking tests were not performed due to the framework's constraints.

Publish Invoice:

The functionality for publishing an invoice was tested by publishing invoice and verifying the correct operation of the feature. As an important example of a constraint we checked that an invoice could not be published having a bigger money quantity than the sponsorship in which he belongs. In addition it was tried that we cannot publish a project that is already published.

4.1.2 Coverage analysis:

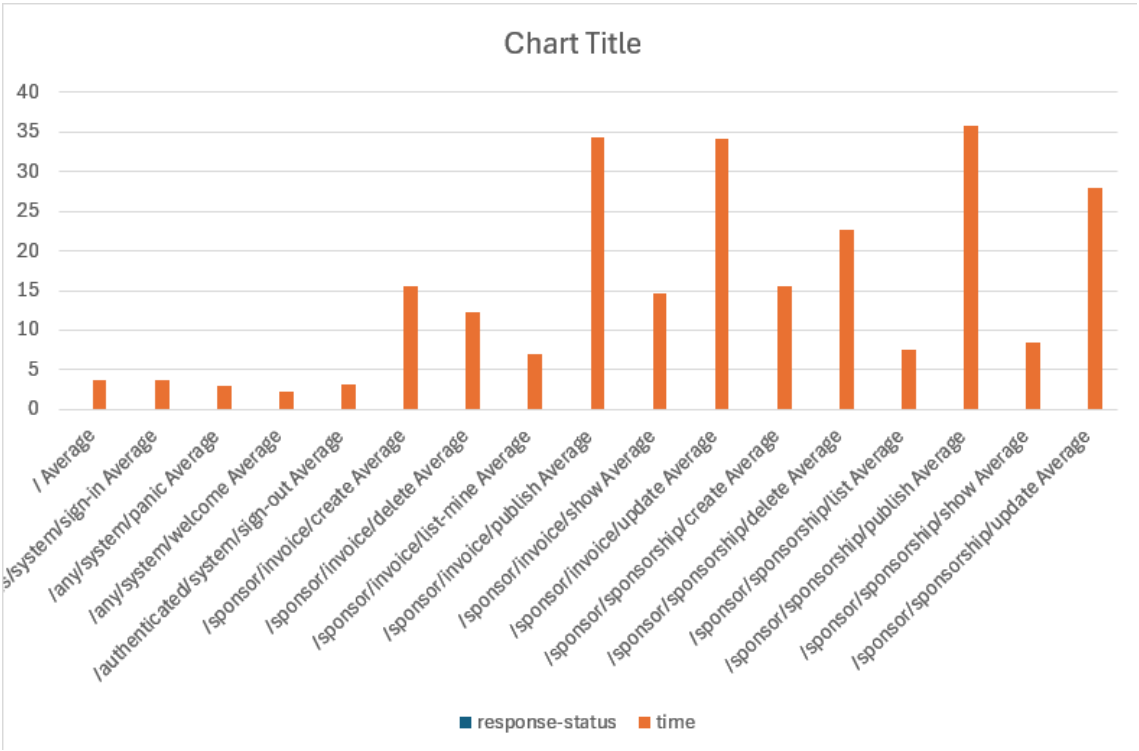
Class Name	Coverage
SponsorInvoiceListSponsorship	96,0%
SponsorInvoiceDeleteService	90,2%
SponsorInvoiceShowService	97,3%
SponsorInvoiceCreateService	94,8%
SponsorInvoiceUpdateService	95,4%
SponsorInvoicePublishService	95,5%

SponsorSponsorshipListService	93,1%
SponsorSponsorshipShowService	97,1%
SponsorSponsorshipDeleteService	94,1%
SponsorSponsorshipCreateService	95,5%
SponsorSponsorshipUpdateService	95,8%
SponsorSponsorshipPublishService	96,7%

Providing the table with the data, we can conclude that the coverage is overall complete knowing that the non-covered code, is mainly code provided by the ACME framework or unreachable.

4.2Performance analysis:

After the execution of the tests, the following chart shows the results of the average performance of each manager related action of the service.



Looking into this graph we can conclude that the most time-consuming action is the publish of the sponsorships, invoices and update of the invoices. This makes sense when analyzing that the feature have a more complex validation than the other ones. Overall, invoices and sponsorship does require approximately the same time.

- Confidence interval:

Let us now see the computations for the confidence interval:

Column1	
Mean	18,28268261
Standard Error	0,667116589
Median	15,679001
Mode	2,6414
Standard Deviation	15,44486565
Sample Variance	238,543875
Kurtosis	0,899597874
Skewness	0,870806282
Range	102,307099
Minimum	1,3758
Maximum	103,682899
Sum	9799,51788
Count	536
Confidence Level(95,0%)	1,310489173

Interval(ms	16,97219	20,92408
	0,016972	0,020924

With the absence of any specific requirement for the confidence interval in our project, a confidence interval of [0,017 , 0,021] assuming the 95% confidence level is in general an acceptable interval time.

- Hypothesis contrast:

We do not have any requirement according to improve or change the performance of the system and indexes are already implemented, then, to simulate a hypothesis contrast let us assume an increase of 10% over the time.

Keeping this in mind, this is the confidence interval of the generated data in contrast.

Mean	20,11095087
Standard Error	0,733828248
Median	17,2469011
Mode	2,90554
Standard Deviation	16,98935222
Sample Variance	262,3982625
Kurtosis	0,899597874
Skewness	0,95788691
Range	112,5378089
Minimum	1,51338
Maximum	114,0511889
Sum	10779,46967
Count	536
Confidence Level(95	1,441538091

Interval(ms	18,66941278	21,55248896

Z-testing analysis:

	103,683	*114,051*
Mean	18,12306	19,93536
Known Var	238,5439	262,3983
Observatic	535	535
Hypothesiz	0	
z	-1,8729	
P(Z<=z) on	0,030541	
z Critical o	1,644854	
P(Z<=z) tw	0,061082	
z Critical tv	1,959964	

In order to measure the impact of the change it is a need to pay attention to the "P(Z<=z) two-tail" row. It shows a value of 0,06 with this is a value in control in order to make comparison knowing that α = 1 - confidence level = 0,05, Since P(z<=z) is above 0,05 and it is not really close to 0,05 we can see that the changes did not affect to our performance, the sample times are different but globally they are the same according to the application performance.

5. Conclusions

As a conclusion, the thorough testing and performance analysis conducted on the mandatory features developed by Student #4 provide valuable insights into the efficiency of the system.

Through examination of test cases, we have ensured that critical functionalities such as sponsorship and their correlated invoices management meet the required standards of correctness and security.

In the performance area we have identified key areas of optimization. The observed variations in performance metrics, with the computation of confidence intervals and hypothesis contrasts, offer a huge understanding of the system's efficiency under different scenarios.

6. Bibliography

Intentionally blank