27 DE MAYO DE 2024

TESTING REPORT STUDENT #3

GROUP C-1.047

Acme SF, Inc.

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

María C. Rodríguez Millán
Email: marrodmil@alum.us.es

Table of contents

1. Executive summary

The contents of this testing report present a testing analysis and a performance analysis. The testing analysis includes two subsections; the test case analysis, which delves into the test cases carried out, and the coverage analysis, which delves into the coverage achieved through the testing. The performance analysis includes, in addition to some performance data, some details about the confidence interval, which were computed with 95% confidence intervals, and the hypothesis contrast, which presents a variation of 10% in time.

2. Revision table

Revision number	Date	Description
1.0.0	27-05-2024	Creation and development of the document.

3. Introduction

The content of this document aims to redact, through its sections, different data crucial in order to understand properly the functioning and efficiency of the system.

The first mentioned section focuses on the test cases analysis, with a detailed description of how each of them were carried out. It must be considered that in each of the test cases analysis, some files (where the recording of the tests are allocated) are mentioned. These files have the same names for both features of the training modules and the training sessions, but can be

found in their corresponding folders (training-module or training-session, respectively). Also, after running the tests, there is an analysis of the code coverage reached through those tests.

The second mentioned section focuses on the performance analysis, providing a graph to visually appreciate the time consumed by the different implemented features. Also, it can be found some details about the confidence interval, that were computed assuming a 95% confidence intervals, and an hypothesis contrast, which compares the before and after of the performance with a 10% increase over the time.

This document is structured starting with a cover page containing relevant information about the document, such as its title, the project's repository, the group number, the author or authors' names with their corresponding corporate email addresses, and the date. The document proceeds with a table of contents, an executive summary, and an introduction before delving into its content. The testing report provides two main sections; a test cases analysis and a performance analysis. Each chapter focuses on both analyzing carefully and providing details regarding the system's efficiency and performance in its D04 regarding the features developer by the Student#3. Finally, the document ends with a conclusion summarizing the presented information and an appropriate bibliography. This structure, as described and used, is outlined in the "Annexes" document, provided by the University of Seville.

4. Contents

4.1Testing analysis:

In this section, it will be analyzed the test suite generated for the mandatory features of Student#3. This analysis is divided into two subsections: the first one; the test case analysis, which focuses on delving into each individual test case, and the second one; the coverage analysis, which delves into the coverage reached through these test cases.

4.1.1 Test case analysis:

- List training module:

This test case involves accessing the listing of the training modules of a developer to verify the feature's correctness. This was achieved through the interaction with the listing of training modules from that developer. This was recorded in the "list.safe" file.

There was also another testing related to the listing of the training modules, which involved accessing this listing from various roles not authorized to do so. An error is arisen when trying to do so, which means the feature is correctly implemented. This was recorded in the "list.hack" file.

- Show training module:

This test case involves accessing the details of a developer's training module to verify the feature's correctness. This was achieved through accessing multiple data from different training modules from that developer. This was recorded in the "show.safe" file.

Furthermore, there was another testing related to the showing of the training modules, which involved accessing this showing from various roles not authorized to do so, including the role from the other existing developer. It was also tested the show for an invalid training

module identifier. An error is arisen when trying these actions, which means the feature is correctly implemented. This was recorded in the “show.hack” file.

- Create training module:

This test case involves trying many values in the creating feature of the training modules to verify that the feature is working properly. This was done through exhaustive combinations of data to check every possible conflictive data, generating positive and negative tests and following a methodology of first, sending the form empty, and second, trying valid values, incorrect values and conflictive data (for example, SQL injections or script hacking for the strings). Finally, a form with all correct data is sent to check the correctness of the system handling the action being performed. The system must not panic when handling negative nor conflictive data, which was achieved successfully. This test cases were recorded in the “create.safe” file.

There was not another testing related to the creation of a training module since that kind of testing is done through POST hacking, and POST hacking is not supported currently by the framework (it needs the usage of external tools such as Postman).

- Delete training module:

This test case involves performing the deletion of several training modules of a certain developer to verify the feature’s correctness. Since the “Delete” button is only shown when the training module is in draft mode, the “delete.safe” file only contains positive deletions. The system must not panic when these actions are performed, which was achieved successfully.

Furthermore, some testing was performed to check that no delete action was allowed to published training modules even when having the right role and the right user, since this is a forbidden action. An error is arisen when trying this action, which means the feature is correctly implemented. This was recorded in the “delete.hack file.

- Update training module:

This test case involves trying many values in the updating feature of the training modules to verify that the feature is working properly. This was done through exhaustive combinations of data to check every possible conflictive data, generating positive and negative tests and following a methodology of first, sending the form empty, and second, trying valid values, incorrect values and conflictive values. Finally, a form with all correct data is sent to check the correctness of the system handling the action being performed.

The system must not panic when handling any of these data, which was achieved successfully. Since the “Update” button is only shown when the training module is in draft mode, the “update.safe” file only contains positive updates.

Furthermore, some testing was performed to check that no update action was allowed to published training modules even when having the right role and the right user, since this is a forbidden action. An error is arisen when trying this action, which means the feature is correctly implemented. This was recorded in the “update.hack” file.

- Publish training module:

This test case involves trying many values in the publishing feature of the training modules to verify that the feature is working properly. This was done through exhaustive combinations of data to check every possible conflictive data, generating positive and negative tests and following a methodology of first, sending the form empty, and second, trying valid values, incorrect values and conflictive values. Finally, a form with all correct data is sent to check the correctness of the system handling the action being performed.

The system must not panic when handling any of these data, which was achieved successfully. Since the "Publish" button is only shown when the training module is in draft mode, the "publish.safe" file only contains positive publishings.

Furthermore, some testing was performed to check that no publish action was allowed to already published training modules even when having the right role and the right user, since this is a forbidden action. An error is arisen when trying this action, which means the feature is correctly implemented. This was recorded in the "publish.hack" file.

List-mine training session:

This test case involves accessing the listing of the training sessions of a developer to verify the feature's correctness. This was achieved through the interaction with the listing of training sessions from that developer. This was recorded in the "list-mine.safe" file.

There was also another testing related to the listing of the training sessions, which involved accessing this listing from various roles not authorized to do so. An error is arisen when trying to do so, which means the feature is correctly implemented. This was recorded in the "list-mine.hack" file.

- List training session:

This test case involves accessing the listing of the training sessions of a developer's training module through the button displayed in the showing feature of the training module to verify the feature's correctness. This was achieved through the interaction with the aforementioned button. This was recorded in the "list.safe" file.

There was also another testing related to the listing of the training sessions of a given training module, which involved accessing this listing from various roles not authorized to do so, as well as from the role of the other developer since the request had an associated training module id. An error is arisen when trying to do so, which means the feature is correctly implemented. This was recorded in the "list.hack" file.

- Show training session:

This test case involves accessing the details of a developer's training session to verify the feature's correctness. This was achieved through accessing multiple data from different training sessions from that developer. This was recorded in the "show.safe" file.

Furthermore, there was another testing related to the showing of the training sessions, which involved accessing this showing from various roles not authorized to do so, including the role from the other existing developer. It was also tested the show for an invalid training session identifier. An error is arisen when trying these actions, which means the feature is correctly implemented. This was recorded in the "show.hack" file.

- Create training session:

This test case involves trying many values in the creating feature of the training sessions to verify that the feature is working properly. This was done through exhaustive combinations of data to check every possible conflictive data, generating positive and negative tests and following a methodology of first, sending the form empty, and second, trying valid values, incorrect values and conflictive data (for example, SQL injections or script hacking for the strings). Finally, a form with all correct data is sent to check the correctness of the system handling the action being performed. The system must not panic when handling negative nor conflictive data, which was achieved successfully. These test cases were recorded in the "create.safe" file.

There was not another testing related to the creation of a training session since that kind of testing is done through POST hacking, and POST hacking is not supported currently by the framework (it needs the usage of external tools such as Postman).

- Delete training session:

This test case involves performing the deletion of a training session of a certain developer to verify the feature's correctness. Since the "Delete" button is only shown when the training session is in draft mode, the "delete.safe" file only contains a positive deletion. The system must not panic when this action is performed, which was achieved successfully.

Furthermore, some testing was performed to check that no delete actions were allowed to published training sessions even when having the right role and the right user, since this is a forbidden action. An error is arisen when trying this actions, which means the feature is correctly implemented. This was recorded in the "delete.hack file.

- Update training session:

This test case involves trying many values in the updating feature of the training sessions to verify that the feature is working properly. This was done through exhaustive combinations of data to check every possible conflictive data, generating positive and negative tests and following a methodology of first, sending the form empty, and second, trying valid values, incorrect values and conflictive values. Finally, a form with all correct data is sent to check the correctness of the system handling the action being performed.

The system must not panic when handling any of these data, which was achieved successfully. Since the "Update" button is only shown when the training session is in draft mode, the "update.safe" file only contains a positive update.

Furthermore, some testing was performed to check that no update action was allowed to published training sessions even when having the right role and the right user, since this is a forbidden action. An error is arisen when trying this action, which means the feature is correctly implemented. This was recorded in the "update.hack" file.

- Publish trainig session:

This test case involves trying many values in the publishing feature of the training sessions to verify that the feature is working properly. This was done through exhaustive combinations of data to check every possible conflictive data, generating positive and

negative tests and following a methodology of first, sending the form empty, and second, trying valid values, incorrect values and conflictive values. Finally, a form with all correct data is sent to check the correctness of the system handling the action being performed.

The system must not panic when handling any of these data, which was achieved successfully. Since the "Publish" button is only shown when the training session is in draft mode, the "publish.safe" file only contains a positive publishing.

Furthermore, some testing was performed to check that no publish action was allowed to already published training sessions even when having the right role and the right user, since this is a forbidden action. An error is arisen when trying this action, which means the feature is correctly implemented. This was recorded in the "publish.hack" file.

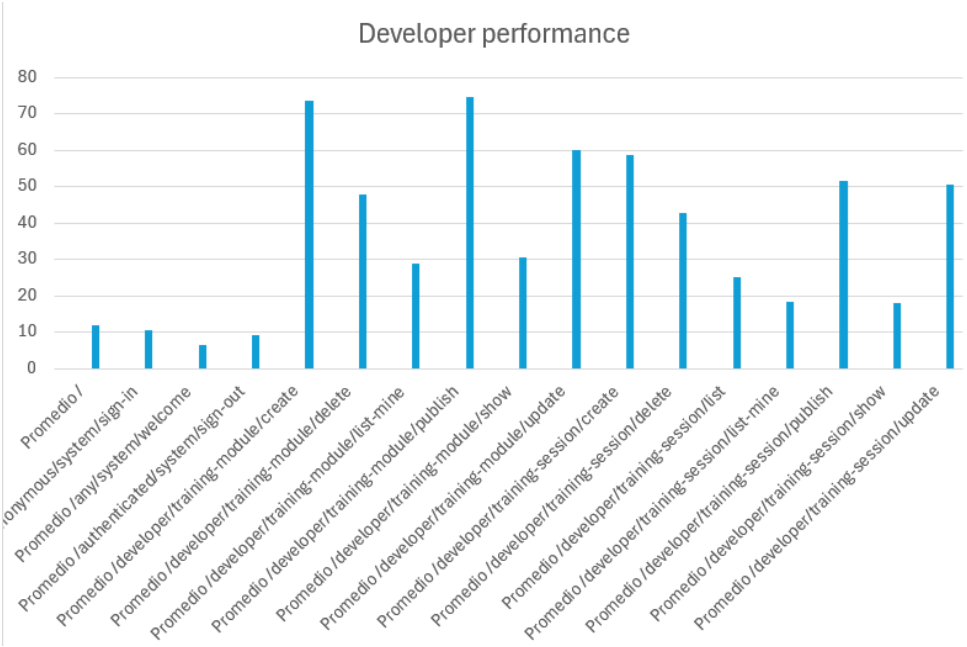
4.1.2 Coverage analysis:

Class Name	Coverage
DeveloperTrainingModuleDeleteService	90,3%
DeveloperTrainingModulePublishService	93,6%
DeveloperTrainingModuleCreateService	92,0%
DeveloperTrainingModuleUpdateService	93,2%
DeveloperTrainingModuleListMineService	94,0%
DeveloperTrainingModuleShowService	97,4%
DeveloperTrainingSessionListService	94,3%
DeveloperTrainingSessionDeleteService	89,7%
DeveloperTrainingSessionPublishService	95,5%
DeveloperTrainingSessionCreateService	95,5%
DeveloperTrainingSessionUpdateService	95,3%
DeveloperTrainingSessionListMineService	93,8%
DeveloperTrainingSessionShowService	95,3%

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:

The following graph formed after the execution of the tests shows the results of the average performance of the developer related actions:



Looking at the graph, it can be seen that the most time-consuming actions are the creation and the publishing of a training module. This is due to the amount of validations and data required by the system to correctly perform these actions. Overall, the create, publish and update actions are the most time-consuming ones due to these mentioned reasons.

- Confidence interval:

The computations for the confidence interval can be found as follows:

Columna1				
		Interval (ms)	43,4552548	46,9775705
Media	45,2164126	Interval (s)	0,04345525	0,04697757
Error típico	0,89746788			
Mediana	45,6332			
Moda	#N/D			
Desviación es	28,2524257			
Varianza de la	798,199555			
Curtosis	16,7855807			
Coeficiente de	1,86092516			
Rango	346,4752			
Mínimo	2,5344			
Máximo	349,0096			
Suma	44809,4649			
Cuenta	991			
Nivel de confia	1,76115785			

Even if we don't have any specific requirement for the confidence interval in our project, a confidence interval of [0,043, 0,046] assuming the 95% confidence level is a starting to become a higher number but it is acceptable since we do not have any limitations in this section.

- Hypothesis contrast:

Since we do not have any requirement regarding the performance of the system and when elaborating this document indexes are already implemented, to simulate a hypothesis contrast it is assumed an increase of 10% over the time.

Taking this into account, the confidence interval of the generated data in contrast can be found as follows:

Before			After		
Media	45,21641261		Media	49,73805387	
Error típico	0,897467878		Error típico	0,987214666	
Mediana	45,6332		Mediana	50,19652	
Moda	#N/D		Moda	#N/D	
Desviación estándar	28,25242565		Desviación estándar	31,07766822	
Varianza de la muestra	798,1995553		Varianza de la muestra	965,8214619	
Curtosis	16,78558067		Curtosis	16,78558067	
Coeficiente de asimetría	1,860925158		Coeficiente de asimetría	1,860925158	
Rango	346,4752		Rango	381,12272	
Mínimo	2,5344		Mínimo	2,78784	
Máximo	349,0096		Máximo	383,91056	
Suma	44809,4649		Suma	49290,41139	
Cuenta	991		Cuenta	991	
Nivel de confianza(95,0%)	1,761157848		Nivel de confianza(95,0%)	1,937273633	
Interval (ms)	43,45525477	46,9775705	Interval (ms)	47,80078024	51,6753275
Interval (s)	0,043455255	0,04697757	Interval (s)	0,04780078	0,05167533

Now, the Z-testing analysis for both cases can be found as follows:

Prueba z para medias de dos muestras		
	Before	*After*
Media	45,21641261	49,73805387
Varianza (conocida)	798,1995553	965,8214619
Observaciones	991	991
Diferencia hipotética de las medias	0	
z	-3,389074009	
P(Z<=z) una cola	0,000350645	
Valor crítico de z (una cola)	1,644853627	
Valor crítico de z (dos colas)	0,000701291	
Valor crítico de z (dos colas)	1,959963985	

Analyzing the impact of the change, it's necessary to pay attention to the "P(Z<=z) two-tail" row. It shows a value of 0,0007, and knowing that α = 1 – confidence level = 0,05, and since it is contained in the interval [0, 0'05], we can say that it has improved.

It seems odd that after increasing the times by 10%, the change has improved the testing. This may be due to the great change in the variance of the sample between both cases.

5. Conclusions

The exhaustive testing and performance analysis carried out in this deliverable of the project have provided crucial data and statistics regarding the efficiency of the system. Through the exhaustive examination of the tests, we can ensure that the functionalities have been correctly tested and are functioning properly in the application.

With the later study of the performance, including factors such as the hypothesis contrast and the confidence interval, we have further details on the system's behavior and efficiency.

6. Bibliography

Intentionally blank