T

Workgroup C1.067

Testing Report

26/05/2025

Miguel Prado Jiménez – [migprajim@alum.us.es](mailto:migprajim@alum.us.es)

<https://github.com/javiarellanoo/Acme-ANS-D04>

Table of Contents

[INTRODUCTION 4](#_Toc199183109)

[FUNCTIONAL TESTING 4](#_Toc199183110)

[Operations on claims by assistance agents 4](#_Toc199183111)

[List 4](#_Toc199183112)

[Show 5](#_Toc199183113)

[Create 5](#_Toc199183114)

[Update 5](#_Toc199183115)

[Publish 6](#_Toc199183116)

[Delete 6](#_Toc199183117)

[Operations on tracking logs by assistance agents 7](#_Toc199183118)

[List 7](#_Toc199183119)

[Show 7](#_Toc199183120)

[Create 8](#_Toc199183121)

[Update 8](#_Toc199183122)

[Publish 9](#_Toc199183123)

[Delete 9](#_Toc199183124)

[PERFORMANCE TESTING 10](#_Toc199183125)

[Mean Confidence Interval 10](#_Toc199183126)

[CONCLUSIONS 12](#_Toc199183127)

[BIBLIOGRAPHY 12](#_Toc199183128)

Executive Summary:

This document presents the formal testing procedure carried out to validate the functionality and performance of the features dealing with the requirements of Student#4, with the aim of assessing complete security within the application, complying with “Ley Orgánica 3/2018”.

The functional testing section presents a description of the test carried out on the features of Assistant Agents, which are claims and tracking logs. In order to carry out this testing, a series of legal and hacking actions have been reproduced. It is also important to mention the focus given to each and every range, valid and invalid values possible for the attributes of each entity.

The second part of this report is about the performance testing of said features, comparing them in two different hardware environments in order to asses which one is more powerful.

Revision Table

|  |  |  |
| --- | --- | --- |
| Revision Number | Date | Description |
| 1.0 | 26/05/2025 |  |
|  |  |  |
|  |  |  |

# INTRODUCTION

The goal of this document is to present an analysis of the testing procedure carried out for requirements #8 and #9 of Student number 4. The testing procedure will be presented in two main parts.

The first part will be the one corresponding to the functional testing. In this section a detailed list with all test cases can be found, each of them with an explanation of how they were carried out. The list will be sorted out by feature.

The second part will contain the information related to the performance testing. It will include charts and information about two different computers in which the testing has been done. A comparative has been made and the result is detailed at the end.

# 

# FUNCTIONAL TESTING

## Operations on claims by assistance agents

### List

The following safe test cases have been carried out:

* Go into the list of claims of every agent, and render every page in order to prove that no problem is encountered.
* Detection of bugs: No bugs were encountered.

The following hack cases have been carried out:

* Try to enter the list of an agent, from a non-authenticated account, and then, from another realm.
* Detection of bugs: No bugs were encountered

### Show

The following safe test cases have been carried out:

* Request the feature to be shown for each and every claim existing in the sample data. Checking that it renders properly
* Detection of bugs: No bugs were encountered.

The following hack cases have been carried out:

* Try to access a claim through the url, but with the value of the claim being null. Afterwards, try again for an existing claim, but through another agent’s account.
* Detection of bugs: No bugs were encountered.
* Try to access an existing claim through the url, but with a non-authenticated account, and the with another realm different from agent.
* Detection of bugs: No bugs were encountered.

### Create

The following safe test cases have been carried out:

* Request the creation of an empty claim. Then, going attribute to attribute try to enter as many valid and invalid data have been considered. These variations were taken from the sample data excel book provided by the lecturers of the subject. Lastly, prove that a correct claim can be created.
* Detection of bugs: No bugs were encountered.

The following hack cases have been carried out:

* Try to create a claim with a null leg associated, then with a leg in draft mode, and finally for a leg published but whose flight was in draft mode.
* Detection of bugs: No bugs were encountered.
* Try to access the creation claim form through the url, but with a non-authenticated account, and the with another realm different from agent.
* Detection of bugs: No bugs were encountered.

### Update

The following safe test cases have been carried out:

* Request the edition of an empty claim. Then, going attribute to attribute try to enter as many valid and invalid data have been considered. These variations were taken from the sample data excel book provided by the lecturers of the subject. Lastly, prove that a correct claim can be updated.
* Detection of bugs: No bugs were encountered.

The following hack cases have been carried out:

* Try to update a claim with null id, then a published claim. Afterwards, try to update a claim with a null leg associated, then with a leg in draft mode, and for a leg published but whose flight was in draft mode. Finally try to access a update form of a claim of another agent.
* Detection of bugs: No bugs were encountered.
* Try to access the update claim form through the url, but with a non-authenticated account, and the with another realm different from agent.
* Detection of bugs: No bugs were encountered.

### Publish

The following safe test cases have been carried out:

* Request the publication of an empty claim. Then, going attribute to attribute try to enter as many valid and invalid data have been considered. These variations were taken from the sample data excel book provided by the lecturers of the subject. Lastly, prove that a correct claim can be published, and prove that a non-completed claim can’t be published.
* Detection of bugs: No bugs were encountered.

The following hack cases have been carried out:

* Try to publish a claim with null id, then a published claim. Afterwards, try to publish a claim with a null leg associated, then with a leg in draft mode, and for a leg published but whose flight was in draft mode. Finally try to access a publish form of a claim of another agent.
* Detection of bugs: No bugs were encountered.
* Try to access the publish claim form through the url, but with a non-authenticated account, and the with another realm different from agent.
* Detection of bugs: No bugs were encountered.

### Delete

The following safe test cases have been carried out:

* Try to delete a claim.
* Detection of bugs: No bugs were encountered.

The following hack cases have been carried out:

* Try to access the delete form (this is implemented because we decided the form not to be available as it is not necessary).
* Detection of bugs: No bugs were encountered.
* Try to access the delete claim form through the url, but with a non-authenticated account, and the with another realm different from agent.
* Detection of bugs: No bugs were encountered.

## Operations on tracking logs by assistance agents

### List

The following safe test cases have been carried out:

* Go into the list of tracking logs of every claim, and render every page in order to prove that no problem is encountered. This means, every claim for every agent, all possible tracking log list
* Detection of bugs: No bugs were encountered.

The following hack cases have been carried out:

* Try to enter the list of tracking logs of a claim of another agent or of a null claim.
* Detection of bugs: No bugs were encountered.
* Try to enter the list of tracking logs of an agent, from a non-authenticated account, and then, from another realm.
* Detection of bugs: No bugs were encountered.

### Show

The following safe test cases have been carried out:

* Request the feature to be shown for each and every tracking log existing in the sample data. Checking that it renders properly
* Detection of bugs: No bugs were encountered.

The following hack cases have been carried out:

* Try to access a tracking log through the url, but with the value of the claim being null. Afterwards, try again for an existing tracking log, but through another agent’s account.
* Detection of bugs: No bugs were encountered.
* Try to access an existing tracking log through the url, but with a non-authenticated account, and the with another realm different from agent.
* Detection of bugs: No bugs were encountered

### Create

The following safe test cases have been carried out:

* Request the creation of an empty tracking log. Then, going attribute to attribute try to enter as many valid and invalid data have been considered. These variations were taken from the sample data excel book provided by the lecturers of the subject. Lastly, prove that a correct tracking log can be created.
* Detection of bugs: No bugs were encountered.
* Some special cases worth mentioning of invalid data are: status must be “PENDING” if the resolution percentage is smaller than 100 and must be different otherwise; If the percentage is 100 the tracking log must have a resolution; The resolution percentage must be greater than the one the last created tracking log has; The tracking log can’t have “DISSATISFACTION” status if the claim is not published.
* Detection of bugs: No bugs were encountered.

The following hack cases have been carried out:

* Try to create a tracking log for a null claim, then for a claim of another agent. Finally, try to access the url of creation for a claim that has already two completed tracking logs (one completed and the exceptional one)
* Detection of bugs: No bugs were encountered.
* Try to access the creation tracking log form through the url, but with a non-authenticated account, and the with another realm different from agent.
* Detection of bugs: No bugs were encountered.

### Update

The following safe test cases have been carried out:

* Request the edition of an empty tracking log. Then, going attribute to attribute try to enter as many valid and invalid data have been considered. These variations were taken from the sample data excel book provided by the lecturers of the subject (and also the ones mentioned in the creation section). Lastly, prove that a correct tracking log can be updated.
* Detection of bugs: No bugs were encountered.

The following hack cases have been carried out:

* Try to update a tracking log with null id, then a published tracking log. Finally try to access a update form of a tracking log of another agent.
* Detection of bugs: No bugs were encountered.
* Try to access the update tracking log form through the url, but with a non-authenticated account, and the with another realm different from agent.
* Detection of bugs: No bugs were encountered.

### Publish

The following safe test cases have been carried out:

* Request the publication of an empty tracking log. Then, going attribute to attribute try to enter as many valid and invalid data have been considered. These variations were taken from the sample data excel book provided by the lecturers of the subject(and also the ones mentioned in the creation section). Lastly, prove that a correct tracking log can be published.
* Detection of bugs: No bugs were encountered.

The following hack cases have been carried out:

* Try to publish a tracking log with null id, then a published tracking log. Finally try to access a publish form of a tracking log of another agent.
* Detection of bugs: No bugs were encountered.
* Try to access the publish tracking log form through the url, but with a non-authenticated account, and the with another realm different from agent.
* Detection of bugs: No bugs were encountered.

### Delete

The following safe test cases have been carried out:

* Try to delete a tracking log.
* Detection of bugs: No bugs were encountered.

The following hack cases have been carried out:

* Try to access the delete form (this is implemented because we decided the form not to be available as it is not necessary).
* Detection of bugs: No bugs were encountered.
* Try to access the delete tracking log form through the url, but with a non-authenticated account, and the with another realm different from agent.
* Detection of bugs: No bugs were encountered.

# PERFORMANCE TESTING

In this section, the performance of the project (regarding the requirements mentioned above) will be evaluated by measuring the time taken to complete the requests done during the functional testing. The objective is to determine if the final product is between the acceptable interval of performance, and which computer performs the best.

To collect the data required, we will run all functional tests for assistance agent’s features. We will be using the following devices:

* HP elite dragonfly: 16 GB RAM, 512 GB SSD memory
* HP 15s-fq4xxx: 16 GB RAM, 500 GB SSD memory

## Mean Confidence Interval

After filtering and cleaning up the data obtained from the trace file obtained after running a replay of the test suites, a analysis has been made following the procedure explained in the slides of the subject to obtain the average time for every request.

For the first computer, we have obtained a grand average of 78.7 ms. As can be appreciated, we see that the MIR is the claim list, whose average is 527.38 ms.

Using the data analyzer extension from Excel, we obtain that the amplitude of the confidence interval at 95% is 11.55 ms. By removing and adding this value from the average, we obtain the confidence interval: [0.075 s – 0.098 s]

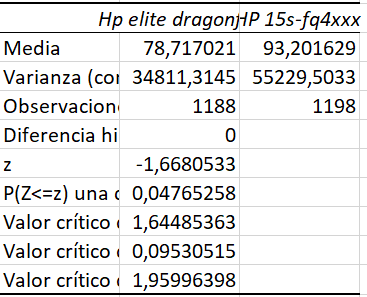
When doing the same with the second computer, we obtained a similar graph. In this case, the grand average goes up to 96.2 ms. As can be appreciated, we see that the MIR continues to be the claim list, whose average is 653.76 ms.

Using the data analyzer, we get that the amplitude of the confidence interval at 95% is 14.54 ms. We would obtain the following confidence interval: [0.088 s – 0.11s]

## Contrasting information

Given the previous results, we can induce that the computer with the best performance is the HP elite dragonfly, as in general terms, all its average times are lower. Let us use the Z-Test with alpha 0.05 (one minus confidence) to verify our hypothesis.

Once the Z-Test has been conducted, the results are the following:



The p-value is 0.09, so no important change is regarded. The first computer is better, but the difference between both is not significant.

# CONCLUSIONS

This document gives a detailed explanation of the test that have been carried out in order to test the requirements of the Student#4. These tests have provided light in order to have a free bug application. Also, testing hack cases has proven that the product we deliver deauthorizes every illegal request discussed in the subject.

Each test case was recorded systematically, grouped by feature and the bugs found, if any, have been reported. This serves as a transparent reference of the testing process.

Through the analysis of performance we have proven that the confidence interval meets the requirements. Which is the most important of the two things assessed in this section. The second one is that, the computer Hp elite dragonfly performs faster, but the difference is not significant.

# BIBLIOGRAPHY

Design & Testing II Slides – S02 Performance Testing