Testing Report – Student #3

* **Group Number**: C1.034
* **Repository**: https://github.com/CarlosCerdaMorales/dp2-c1-034
* **Workgroup Members** **and Corporate Emails:**

Loubna Founoun Elaoud: [loufouela@alum.us.es](mailto:loufouela@alum.us.es)

Carlos Cerdá Morales: [carcermor@alum.us.es](mailto:carcermor@alum.us.es)

José Luis Moraza Vergara: [josmorver@alum.us.es](mailto:josmorver@alum.us.es)

Claudia Oviedo Govantes: [claovigov@alum.us.es](mailto:claovigov@alum.us.es)

Sergio Cantillo Blanco: [sercanbla@alum.us.es](mailto:sercanbla@alum.us.es)

* **Date**: Sunday 25th, May 2025

**Table Of Contents**

[1. Executive Summary 3](#_451ewtno8qvm)

[2. Revision Table 3](#_ov3mb6vwut6t)

[3. Introduction 3](#_r2bem3ub6oc7)

[4. Contents 3](#_nh08ol2sdh4r)

[4.1 Functional Testing 3](#_h69on2nqubkm)

[4.2 Performance Testing 3](#_xn4c25wc84mn)

[5. Conclusions 3](#_2l2wgk3amr06)

[6. Bibliography 3](#_a255vkbpqjez)

# Executive Summary

This document details the functional and performance evaluations I carried out on my student. The primary goal was to confirm that all components operate as intended and to assess the system's responsiveness under standard usage scenarios.

For the functional evaluation, I grouped the test cases according to individual features. Each test focused on a particular function, ensuring that the application's behavior aligns with expected outcomes.

In terms of performance assessment, I adhered to the procedure outlined in the session manual. I gathered execution time data from .trace files and processed the results using Excel. I then created visual representations and computed 95% confidence intervals to determine whether the response times stayed within acceptable parameters. The testing was conducted under two separate setups: one with the database running without supplemental indexes, and another with appropriate indexing enabled. A statistical analysis was performed to compare both configurations and quantify the performance improvements resulting from indexing.

To conclude, this report summarizes the testing efforts on the application’s core functionalities, supported by performance metrics that offer meaningful insight into the system’s behavior in practical, real-world conditions

# Revision Table

|  |  |  |
| --- | --- | --- |
| **Revision Number** | **Date** | **Description** |
| 1.0 | 25/05/2025 | Final version |

# Introduction

This report presents the testing process I undertook for my project, addressing both functional accuracy and system performance. The main objective was to verify that all features function properly and that the system responds efficiently under typical usage conditions.

I began with functional testing, organizing the test cases by feature to ensure each part of the application performed as expected. Once functionality was confirmed, I shifted focus to performance testing, where I measured how quickly the system handled different types of requests. This analysis helped uncover potential slowdowns and provided a basis for comparing performance between different system configurations.

Through this testing process, I gained a deeper understanding of the system's behavior and reliability in scenarios that closely resemble real-world use.

# Contents

## 4.1 Functional testing

During the testing phase, I focused on verifying the features in *Acme-ANS* designed for flight crew members, particularly those involving the tracking of flight assignments and activity history. The outcomes of these tests were captured in trace files, which are stored in the following directories: /src/test/resources/flight-crew-member/flight-assignment and /src/test/resources/flight-crew-member/activity-log. The test suite achieved full coverage across all cases, as evidenced by the image included below.

Imagen que contiene Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

|  |  |
| --- | --- |
| Flight CReW member flight assignment features | |
| CREATE SERVICE SAFE | |
| TC-01: CREATE AN EMPTY assignment | The system appropriately displayed error messages when an attempt was made to create an assignment without any data, correctly identifying that specific fields were required and "must not be null." |
| TC-02: INVALID Remarks | A text that exceeded the 255 characters was submitted as the remarks, resulting in a rejected attempt to create the assignment. |
| TC-03: VALID REMARKS | Differents type of texts were submitted to the remarks field, all within the 255 characters. Non-latin texts and injections were also submited. All attempts were successful. |
| TC-04: VALID Flight CREW DUTY | All possible values of the flight crew duty were tried and successfully accepted by the system. |
| TC-05: VALID ASSIGNMENT STATUS | All possible values of the assignment status were tried and successfully accepted by the system. |
| TC-06: VALID LEG | A lot of different legs were submitted to the field. All of them were successful. |
| TC-07: INVALID FLIGHT CREW DUTY | An attempt was made to create an assignment without assigning any flight crew duty, resulting in an error indicating that the duty field must not be null. |
| TC-08: INVALID ASSIGNMENT STATUS | An attempt was made to create an assignment without assigning any status, resulting in an error indicating that the status field must not be null. |
| TC-09: INVALID LEG | An attempt was made to create an assignment without assigning any leg, resulting in an error indicating that the leg field must not be null. |
| BUGS | No bugs were found while recording these tests. |
| CREATE SERVICE HACK | |
| TC-10: INVALID ID FOR CREATE | There was an attempt to alter the id field of a newly created assignment in an effort to overwrite an existing record in the database, rather than generate a new entry. The application correctly rejected this action, returning an "Access Unauthorised" response. |
| TC-11: INVALID FLIGHT CREW DUTY | Attempts were made to modify the flight crew duty field with values not accepted by the enumeration, including numbers, empty strings, and other invalid entries. All these attempts resulted in an "Access Unauthorised" response from the application. |
| TC-12: INVALID assignment status | Attempts were made to modify the assignment status field with values not accepted by the enumeration, including numbers, empty strings, and other invalid entries. All these attempts resulted in an "Access Unauthorised" response from the application. |
| TC-13: INVALID LEG | An attempt was made to include a leg in the assignment that was either unpublished or did not belong to the company associated with the logged-in crew member. In both cases, the application responded with "Access Unauthorised." This same response was consistently returned when trying to assign a leg that was not found in the database, indicating uniform handling of unauthorized or invalid references. |
| BUGS | No bugs were found while recording these tests. |
| DELETE SERVICE SAFE | |
| TC-14: DELETE assignment | An attempt to delete an assignment was carried out. Because the sole requirement is that the assignment belongs to the user, the system successfully completed the deletion after confirming this condition was met. |
| BUGS | No bugs were found while recording these tests. |
| DELETE SERVICE HACK | |
| TC-15: DELETE NON-EXISTENT assignment | A deletion request was made for an assignment that does not exist, by calling the /delete endpoint with an invalid ID. The application properly managed the situation and returned an "Access Unauthorised" error in response. |
| TC-16: DELETE Assignment NOT MINE | A deletion attempt was made on an assignment that belonged to another user by sending a request to the /delete endpoint with a valid assignment ID. The system accurately detected the unauthorized access and returned an "Access Unauthorised" error. |
| TC-17: DELETE PUBLISHED assignment | An attempt to delete an assignment owned by the current user was made; however, since the assignment was already published, the application properly denied the request. It responded with an "Access Unauthorised" error, enforcing the rule that published assignments cannot be deleted. |
| BUGS | No bugs were found while recording these tests. |
| LIST-completed SERVICE SAFE | |
| TC-18: LIST completed flight assignments | A request was made to list assignments associated with a leg that has already concluded. This operation consistently succeeded because the database query filters results using the ID of the logged-in user. |
| BUGS | No bugs were found while recording these tests. |
| LIST-completed SERVICE HACK | |
| THERE IS NO hacking attempt applicable to this specific FEATURE. | |
| LIST-UNCOMPLETED SERVICE SAFE | |
| TC-19: LIST completed flight assignments | A request was made to list assignments associated with a leg that has not yet concluded. This operation consistently succeeded because the database query filters results using the ID of the logged-in user. |
| BUGS | No bugs were found while recording these tests. |
| LIST-UNCOMPLETED SERVICE HACK | |
| THERE IS NO hacking attempt applicable to this specific FEATURE. | |
| PUBLISH SERVICE SAFE | |
| TC-20: PUBLISH an empty ASSIGNMENT | The system appropriately displayed error messages when an attempt was made to publish an assignment without any data, correctly identifying that specific fields were required and "must not be null." |
| TC-21: INVALID remarks | A text that exceeded the 255 characters was submitted as the remarks, resulting in a rejected attempt to create the assignment. |
| TC-22: valid remarks | Differents type of texts were submitted to the remarks field, all within the 255 characters. Non-latin texts and injections were also submited. All attempts were successful. |
| TC-23: VALID flight crew duty | All possible values of the flight crew duty were tried and successfully accepted by the system. |
| TC-24: VALID assignment status | All possible values of the assignment status were tried and successfully accepted by the system. |
| TC-25: inVALID flight crew duty | An attempt was made to create a assignment without assigning any flight crew duty, resulting in an error indicating that the duty field must not be null. When the leg selected already had a pilot or a copilot, we tried submitting the pilot and copilot duty, respectively. Both attemps failed because you can not publish an assignment with a leg that already has a pilot or copilot with the pilot and copilot duty. |
| TC-26: valid leg | Attempts were made to store the lastNibble field with values breaking the pattern by exceeding upper and lower limits, as well as including letters and non-numeric characters. These attempts were rejected by the database. |
| TC-27: invalid leg | An attempt was made to create a assignment without assigning any leg, resulting in an error indicating that the leg field must not be null. A leg that had been already completed was also submitted, as well as a leg that overlapped with the assignments that the member already had. Both attempts were unsuccessfull as you can only publish assignments with legs that are uncompleted and that do not overlap with other assignments. |
| Tc-28: invalid assignment status | An attempt was made to create an assignment without assigning any status, resulting in an error indicating that the status field must not be null. |
| TC-29: VALID PUBLISH | Finally, an assignment was successfully published after ensuring that all required fields were valid and properly completed, and all restrictions were checked. |
| BUGS | No bugs were encountered while recording tests. |
| PUBLISH SERVICE hack | |
| TC-30: PUBLISH NON-EXISTENT ID | An attempt was made to access the /publish URI with an assignment ID that does not exist. The system correctly handles it with an "access unauthorised" response. |
| TC-31: PUBLISH ALREADY PUBLISHED | An attempt was made to access the /publish URI with the ID of an already published assignment, resulting in an "unauthorised" response. |
| TC-32: PUBLISH NOT MINE | An attempt was made to publish an assignment that does not belong to the client in session, resulting in an authorisation error. |
| TC-33: PUBLISH INVALID FLIGHT CREW DUTY | Attempts were made to modify the flight crew duty field with values not accepted by the enumeration, including numbers, empty strings, and other invalid entries. All these attempts resulted in an "Access Unauthorised" response from the application. |
| TC-34: PUBLISH INVALID ASSIGNMENT STATUS | Attempts were made to modify the assignment status field with values not accepted by the enumeration, including numbers, empty strings, and other invalid entries. All these attempts resulted in an "Access Unauthorised" response from the application. |
| TC-35: PUBLISH INVALID LEG | An attempt was made to include a leg in the assignment that was either unpublished or did not belong to the company associated with the logged-in crew member. In both cases, the application responded with "Access Unauthorised." This same response was consistently returned when trying to assign a leg that was not found in the database, indicating uniform handling of unauthorized or invalid references. |
| BUGS | No bugs were found while recording these tests. |
| SHOW SERVICE SAFE | |
| TC-36: SHOW VALID ASSIGNMENT | An attempt was made to display a valid assignment, as it belongs to the current client. |
| BUGS | No bugs were found while recording these tests. |
| show service hack | |
| TC-37: SHOW NON-EXISTENT ASSIgnment | An attempt was made to display an assignment that does not exist in the database, resulting in Access Unauthorised. |
| TC-38: show other member’s unpublished assignment | An attempt was made to display another unpuplished member's assignment, resulting in Access Unauthorisede. |
| BUGS | No bugs were found while recording these tests. |
| UPDATE SERVICE SAFE | |
| TC-39: UPDATE AN EMPTY assignment | The system appropriately displayed error messages when an attempt was made to update an assignment without any data, correctly identifying that specific fields were required and "must not be null." |
| TC-40: INVALID remarks | A text that exceeded the 255 characters was submitted as the remarks, resulting in a rejected attempt to update the assignment. |
| TC-41: valid remarks | Differents type of texts were submitted to the remarks field, all within the 255 characters. Non-latin texts and injections were also submited. All attempts were successful. |
| TC-42: VALID flight crew duty | All possible values of the flight crew duty were tried and successfully accepted by the system. |
| TC-43: VALID assignment status | All possible values of the assignment status were tried and successfully accepted by the system. |
| TC-44: VALID leg | A lot of different legs were submitted to the field. All of them were successful. |
| TC-45: INVALID flight crew duty | An attempt was made to create an assignment without assigning any flight crew duty, resulting in an error indicating that the duty field must not be null. |
| TC-46: invalid assignment status | An attempt was made to create an assignment without assigning any status, resulting in an error indicating that the status field must not be null. |
| TC-47: INVALID leg | An attempt was made to create an assignment without assigning any leg, resulting in an error indicating that the leg field must not be null. |
| BUGS | No bugs were found while recording these tests. |
| UPDATE SERVICE hack | |
| TC-48: UPDATE NON-EXISTENT ID | An attempt was made to access the /update URI with an assignment ID that does not exist. The system correctly handles it with an "access unauthorised" response. |
| TC-49: UPDATE ALREADY PUBLISHED | An attempt was made to access the /update URI with the ID of an already published assignment, resulting in an "unauthorised" response. |
| tc-50: UPDATE not mine | An attempt was made to update an assignment that does not belong to the member in session, resulting in an authorisation error. |
| tc-51: UPDATE invalid flight crew duty | Attempts were made to modify the flight crew duty field with values not accepted by the enumeration, including numbers, empty strings, and other invalid entries. All these attempts resulted in an "Access Unauthorised" response from the application. |
| TC-52: UPDATE INVALID ASSIGNMENT STATUS | Attempts were made to modify the assignment status field with values not accepted by the enumeration, including numbers, empty strings, and other invalid entries. All these attempts resulted in an "Access Unauthorised" response from the application. |
| TC-53: UPDATE INVALID leg | An attempt was made to include a leg in the assignment that was either unpublished or did not belong to the company associated with the logged-in crew member. In both cases, the application responded with "Access Unauthorised." This same response was consistently returned when trying to assign a leg that was not found in the database, indicating uniform handling of unauthorized or invalid references |
| BUGS | No bugs were found while recording these tests. |

|  |  |
| --- | --- |
| FLight crew member activity log features | |
| CREATE SERVICE SAFE | |
| TC-54: CREATE AN EMPTY ACTIVITy log | When attempting to create an activity log without providing any field values, the application returned multiple validation warnings, indicating that certain fields "may not be null." |
| TC-55: INVALID INCIDENT TYPE | An attempt was made to create a log entry using an incident type that violated the defined upper and lower boundary constraints. |
| TC-56: VALID incident type | A log entry was successfully created using an incident type that fell within the allowed range and included non-Latin and injections characters, all without any errors. |
| TC-57: INVALID DESCRIPTION | An attempt was made to create a log entry using a description that violated the defined upper and lower boundary constraints. |
| TC-58: VALID description | A log entry was successfully created using a description that fell within the allowed range and included non-Latin and injections characters, all without any errors. |
| TC-59: INVALID severity level | An attempt was made to create a log entry using an severity level that violated the defined upper and lower boundary constraints, as well as numbers with decimals and strings that were unvalid. |
| TC-60: Valid severity level | Integer values within the range were tried. They were all successful. |
| BUGS | No bugs were found while recording these tests. |
| CREATE SERVICE HACK | |
| TC-61: INVALID ID FOR CREATE | An attempt was made to modify the id field of the new log being created, aiming to overwrite an existing log in the database instead of creating a new one. This attempt resulted in an "Access Unauthorised" response from the application. |
| BUGS | No bugs were found while recording these tests. |
| DELETE SERVICE SAFE | |
| TC-62: DELETE passenger | An attempt was made to delete a log. Since the only restriction is that it must be the user’s own log, the deletion was performed successfully after verifying this condition. |
| BUGS | No bugs were found while recording these tests. |
| DELETE SERVICE HACK | |
| TC-63: DELETE NON-EXISTENT log | An attempt was made to delete a non-existent log by accessing the /delete endpoint with an ID that does not exist in the database. The application correctly handled the request and responded with an "Access Unauthorised" error. |
| TC-64: DELETE log NOT MINE | An attempt was made to delete a log that did not belong to the current user by accessing the /delete endpoint with a valid ID. The system correctly identified the unauthorized action and responded with an "Access Unauthorised" error. |
| TC-65: DELETE PUBLISHED log | An attempt was made to delete a log that belonged to the current user but was already published. The application correctly rejected the operation, responding with an "Access Unauthorised" error, as published passengers cannot be deleted. |
| BUGS | No bugs were found while recording these tests. |
| LIST SERVICE SAFE | |
| TC-66: LIST A flight ASSIGNMENT’s logs | An attempt was made to list the logs of an assignment. This operation was always successful, as the database query is executed based on the ID of the currently logged-in user. |
| BUGS | No bugs were found while recording these tests. |
| LIST SERVICE HACK | |
| TC-67: LIST logs FROM NON-EXISTENT assignment | An attempt was made to list the logs of a non-existent assignment, resulting in Access Unauthorised from the application. |
| TC-68: LIST LOgs FROM unpublished assignment | An attempt was made to list the logs of an unpublished assignment, resulting in Access Unauthorised. |
| BUGS | No bugs were found while recording these tests. |
| PUBLISH SERVICE SAFE | |
| TC-69: PUBLISH an empty log | When attempting to publish an activity log without providing any field values, the application returned multiple validation warnings, indicating that certain fields "may not be null." |
| TC-70: INVALID incident type | An attempt was made to publish a log entry using an incident type that violated the defined upper and lower boundary constraints. |
| TC-71: VALID incident type | A log entry was successfully published using an incident type that fell within the allowed range and included non-Latin and injections characters, all without any errors. |
| TC-72: INVALID description | An attempt was made to publish a log entry using a description that violated the defined upper and lower boundary constraints. |
| TC-73: VALID description | A log entry was successfully published using a description that fell within the allowed range and included non-Latin and injections characters, all without any errors. |
| TC-74: INVALID severity level | An attempt was made to publish a log entry using a severity level that violated the defined upper and lower boundary constraints, as well as numbers with decimals and strings that were unvalid. |
| TC-75: VALID severity level | Integer values within the range were tried. They were all successful. |
| TC-76: inVALID assignment | An unsuccessful attempt was made to publish the log of a flight assignment that had a leg that had not yet landed |
| TC-77: VALID assignment | A successful attempt was made to publish the log of a flight assignment that had a leg that had already landed. |
| BUGS | No bugs were found while recording these tests. |
| PUBLISH SERVICE hack | |
| TC-78: publish NON-EXISTENT log | An attempt was made to publish a non-existent log by accessing the /publish endpoint with an ID that does not exist in the database. The application correctly handled the request and responded with an "Access Unauthorised" error. |
| TC-79: publish log NOT MINE | An attempt was made to publish a log that did not belong to the current user by accessing the /publish endpoint with a valid ID. The system correctly identified the unauthorized action and responded with an "Access Unauthorised" error. |
| TC-80: publish PUBLISHED log | An attempt was made to delete a log that belonged to the current user but was already published. The application correctly rejected the operation, responding with an "Access Unauthorised" error, as published passengers cannot be published. |
| BUGS | No bugs were found while recording these tests. |
| SHOW SERVICE SAFE | |
| TC-81: SHOW VALID log | An attempt was made to display a log, as it has been published or it was unpublished but belonged to the user logged in. |
| BUGS | No bugs were found while recording these tests. |
| show service hack | |
| tc-82: SHOW NON-EXISTENT log | An attempt was made to display a log that does not exist in the database, resulting in Access Unauthorised. |
| tc-83: SHOW other member’s unpublished log | An attempt was made to display another unpublished member's log, resulting in Access Unauthorised. |
| UPDATE SERVICE SAFE | |
| TC-84: CREATE AN EMPTY log | An attempt was made to update a log without filling in the fields, resulting in several "may not be null" warnings. |
| TC-85: INVALID incident type | An attempt was made to update a log entry using an incident type that violated the defined upper and lower boundary constraints. |
| TC-86: VALID incident type | A log entry was successfully update using an incident type that fell within the allowed range and included non-Latin and injections characters, all without any errors. |
| TC-87: INVALID description | An attempt was made to update a log entry using a description that violated the defined upper and lower boundary constraint |
| TC-88: VALID description | A log entry was successfully update using a description that fell within the allowed range and included non-Latin and injections characters, all without any errors. |
| TC-89: INVALID severity level | An attempt was made to update a log entry using a severity level that violated the defined upper and lower boundary constraints, as well as numbers with decimals and strings that were unvalid. |
| TC-90: valid severity level | Integer values within the range were tried. They were all successful. |
| BUGS | No bugs were found while recording these tests. |
| UPDATE SERVICE hack | |
| TC-91: update NON-EXISTENT log | An attempt was made to update a non-existent p by accessing the /update endpoint with an ID that does not exist in the database. The application correctly handled the request and responded with an "Access Unauthorised" error. |
| TC-99: update passenger NOT MINE | An attempt was made to update a passenger that did not belong to the current user by accessing the /update endpoint with a valid ID. The system correctly identified the unauthorized action and responded with an "Access Unauthorised" error. |
| TC-100: update PUBLISHED passenger | An attempt was made to update a passenger that belonged to the current user but was already update. The application correctly rejected the operation, responding with an "Access Unauthorised" error, as published passengers cannot be updated. |
| BUGS | No bugs were found while recording these tests. |

## 4.2 Performance testing

Performance testing was conducted on the same computer. For this purpose, all the previously shown traces were executed twice: first without database indexes, and a second time with indexes, aiming to improve performance by reducing the query resolution time. Below, the timing tables of both analyses will be shown:

Un conjunto de letras blancas en un fondo blanco

El contenido generado por IA puede ser incorrecto.

Un conjunto de letras blancas en un fondo blanco

El contenido generado por IA puede ser incorrecto.

**Conclusion on the Impact of Indexes on Performance**

In this timing analysis conducted on two instances — one without indexes and another with indexes — it was observed that the presence of indexes did not improve performance but instead caused a slight delay in the queries. This can be attributed to several reasons:

1. **Small size of the test data:**  
   In scenarios with a small amount of data, indexes may not be effective because the cost of maintaining and querying the index can outweigh the benefit of avoiding a full table scan. In small databases, a full scan is usually fast and straightforward.
2. **Additional cost of index maintenance:**  
   Creating and updating indexes adds overhead to the database, especially during insert, update, or delete operations. In tests where many similar queries are executed or frequent changes occur, this overhead can be reflected in longer response times.
3. **Specific characteristics of the queries and indexes:**  
   Not all indexes benefit all queries. If queries are not optimized to leverage existing indexes, or if the indexes do not cover the appropriate columns, the database engine may not efficiently use them.

After processing and cleaning the collected data, a descriptive statistical analysis was performed on both traces, gathering various data shown below:

Tabla, Excel

El contenido generado por IA puede ser incorrecto.

Tabla, Excel

El contenido generado por IA puede ser incorrecto.

Across both test configurations, the confidence intervals were very similar and closely clustered, indicating a high degree of consistency in the results. However, there was a slight, recurring pattern in which queries executed without indexes took a bit longer than those without. This suggests that, within the context of this analysis, the use of indexes may introduce a small performance improvement.

To explore this further, a z-value was computed to statistically evaluate the difference between the two sets of execution traces. The results showed statistically significant improvement in query performance when indexes were enabled.

Overall, the study indicates that, under these specific testing conditions, the addition of indexes does provide a measurable performance gain and may even result in a slight decrease in efficiency.

Tabla

El contenido generado por IA puede ser incorrecto.

It can be clearly observed that the average execution time in the second analysis, which includes indexes, is lower than in the first analysis without indexes. This indicates that adding indexes did improve the performance of the testing process.

# Conclusions

The performance analysis comparing query execution times with and without indexes demonstrated that, under the tested conditions, the inclusion of indexes led to a noticeable improvement. The reduction in response times suggests that, even with a relatively small dataset, the query patterns were well-suited to benefit from indexing. Both the descriptive statistics and the calculated z-value supported this outcome, showing that queries executed with indexes generally performed faster, indicating the efficiency gained through optimized data access. To gain a more comprehensive understanding of indexing benefits, further testing with larger datasets and a broader range of query types is recommended.

# Bibliography

Intentionally blank.