Testing Report

* **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**: Day 26th, May 2025

**Table Of Contents**

[1. Executive Summary 3](#_j0ecp5q1msik)

[2. Revision Table 3](#_oe67qpgp37ft)

[3. Introduction 3](#_rq9jqjmldy1c)

[4. Contents 3](#_blq207wmerm)

[4.1 Functional testing 3](#_50b13qjkcmyp)

[4.2 Título 2 7](#_pienl2lpdmgf)

[5. Conclusions 7](#_jzbwog49smxw)

[6. Bibliography 7](#_dt0y38l41do4)

# Executive Summary

In this report, I’ll talk about the functional and performance tests I did on my student project. The main goal was to check that everything works like it’s supposed to and to see how quickly the system responds under normal use.

For the functional tests, I organized the test cases by feature. Each test looked at a specific part of the app to make sure it behaves correctly.

For the performance tests, I gathered data on execution times before and after adding indexes. Then, I created charts and calculated confidence intervals to check if the app’s speed stays within acceptable ranges. I also did a statistical comparison between both sets of results to see how the indexes affected performance.

Overall, this report shows that I tested the main features carefully, and we now have performance data that gives a good idea of how the system runs in real life.

# Revision Table

| **Revision Number** | **Date** | **Description** |
| --- | --- | --- |
| 1.0 | Sun 25, May 2025 | First draft |
| 2.0 | Mon 26, May 2025 |  |

# Introduction

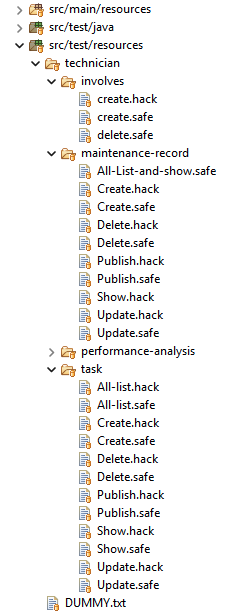
This report aims to describe the testing I performed on my project, focusing on both functionality and performance. The goal was to verify that every feature works properly and that the system responds efficiently to requests.

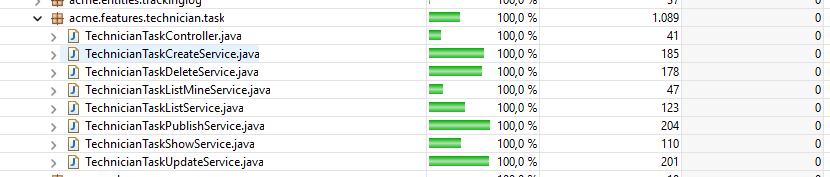
First, I carried out a series of functional tests organized by feature to ensure the app behaved as expected. After that, I concentrated on performance testing by measuring how long the system took to handle various requests. This helped identify any slow points and allowed me to compare performance on different configurations, in my case, applying indexes.

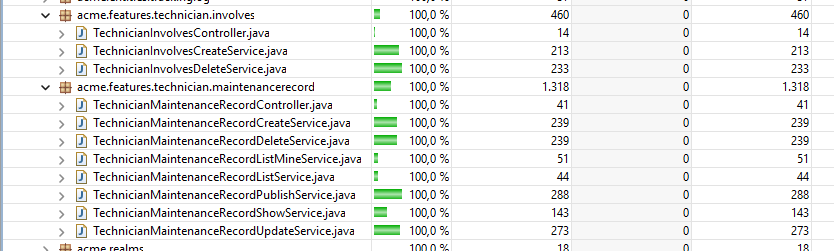
In summary, this testing helped me better understand how the system performs in practice.

# Contents

## 4.1 Functional testing

In these tests, I reviewed the features of my project, Acme-ANS, that relate to technician tasks and maintenance records management. The tests are recorded in trace files found in the /src/test/resources/technician/maintenance-record and /src/test/resources/technician/task. I also tested the intermediate entity between the called “Involves” in /src/test/resources/technician/involves. Every test case reached full coverage, which you can see in the image below.





These are the trace files that demonstrate the code coverage, included just below, and created following the recommendations provided in the course. I tested both *.safe* traces—covering positive and negative test cases—and *.hack* traces, which simulate attempts to hack the application. All hacking attempts were correctly handled and resulted in an "Access Unauthorised" response.

In the following sections, the specific test cases are specified and how each was addressed

**NOTE:** When it says “Each variation of every form field was tested,” it means that for each field, tests included empty values, boundary values at both upper and lower limits (for example, maximum or minimum length or integer value), valid and invalid values by format (such as dates and money type), or by meaning (for example, whether past dates are actually in the past, same for future dates). All cases provided in the *“SampleData”* have been checked.

**Technician task**

| **.safe** | **Description** | **Detected bugs** |
| --- | --- | --- |
| list | - The technicians' tasks were listed, including their own tasks (which may be unpublished) as well as the catalog of tasks published by all technicians. Additionally, tests were done to list tasks linked to specific maintenance records, both published and unpublished (for unpublished ones the list is shown only if you are the owner of the maintenance record). | No bugs were found. |
| show | - Published tasks from the logged-in technician and from other technicians were displayed. - Unpublished tasks from the logged-in technician were also shown. | No bugs were found. |
| create | - An attempt was made to submit an empty form. - Each variation of every form field was tested. | No bugs were found. |
| update | For unpublished tasks of the logged-in technician: - Each variation of every field in the task update form was tested. | No bugs were found. |
| publish | For unpublished tasks of the logged-in technician: - Every variation of each field in the task update form was tested. | No bugs were found. |
| delete | - An attempt was made to delete unpublished tasks of the logged-in technician. | No bugs were found. |
| **. hack** | **Descripción** | **Detected bugs** |
| list | **Incorrect role:** - Tasks were listed without a role or with an incorrect role (both the personal list and the full catalog of published tasks were tested). | No bugs were found. |
| show | **Incorrect role:** - An attempt was made to make the request without an id parameter - Attempts were made to display tasks with ids that do not exist in the database - Attempts were made to display both unpublished and published tasks  **With technician role:** - An attempt was made to make the request without an id parameter - Attempts were made to display tasks with ids that do not exist in the database - Attempts were made to display unpublished tasks NOT assigned to the logged-in technician | No bugs were found. |
| create | **Incorrect role:** - An attempt was made to access the task creation screen.  **With technician role:** - Attempts were made to hack the id by using values that do not exist in the database, including other published and unpublished tasks of the logged-in technician and also tasks from another technician. - Attempts were made to set the status field to invalid values such as 9999, null, or strings that are not part of the enum value set. | It was necessary to modify the authorise method to adjust the validation of the TaskType enum, and unreachable code was removed. |
| update | **Incorrect role:** - An attempt was made to make the request without an id parameter - An attempt was made to make the request with an id that does not exist in the database - Attempts were made to update both published and unpublished tasks  **With technician role:** - An attempt was made to make the request without an id parameter - Update attempts were made on tasks with non-existent ids - Attempts were made to update unpublished tasks of another technician - Attempts were made to update published tasks of the logged-in technician - Attempts were made to update published tasks of another technician - Attempts were made to assign invalid values to the Type field, such as 9999, null, or strings not included in the enum values | It was necessary to modify the authorise method to adjust the validation of the TaskType enum, and unreachable code was removed. |
| publish | **Incorrect role:** - An attempt was made to make the request without an id parameter - An attempt was made to make the request with an id that does not exist in the database - Attempts were made to publish both published and unpublished tasks  **With technician role:** - An attempt was made to make the request without an id parameter - Attempts were made to publish tasks with non-existent ids - Attempts were made to publish unpublished tasks of another technician - Attempts were made to publish published tasks of the logged-in technician - Attempts were made to publish published tasks of another technician - Attempts were made to assign invalid values to the Type field, such as 9999, null, or strings not included in the enum values | It was necessary to modify the authorise method to adjust the validation of the TaskType enum, and unreachable code was removed. |
| delete | **Incorrect role:** - An attempt was made to make the request without an id parameter - Attempts were made to delete published, unpublished, and non-existent records  **With technician role:** - Attempts were made to delete tasks with non-existent ids. - Attempts were made to delete published tasks belonging to both the logged-in technician and other technicians. - Attempts were made to delete existing unpublished tasks of other technicians. | No bugs were found. |

**Technician Maintenance Record**

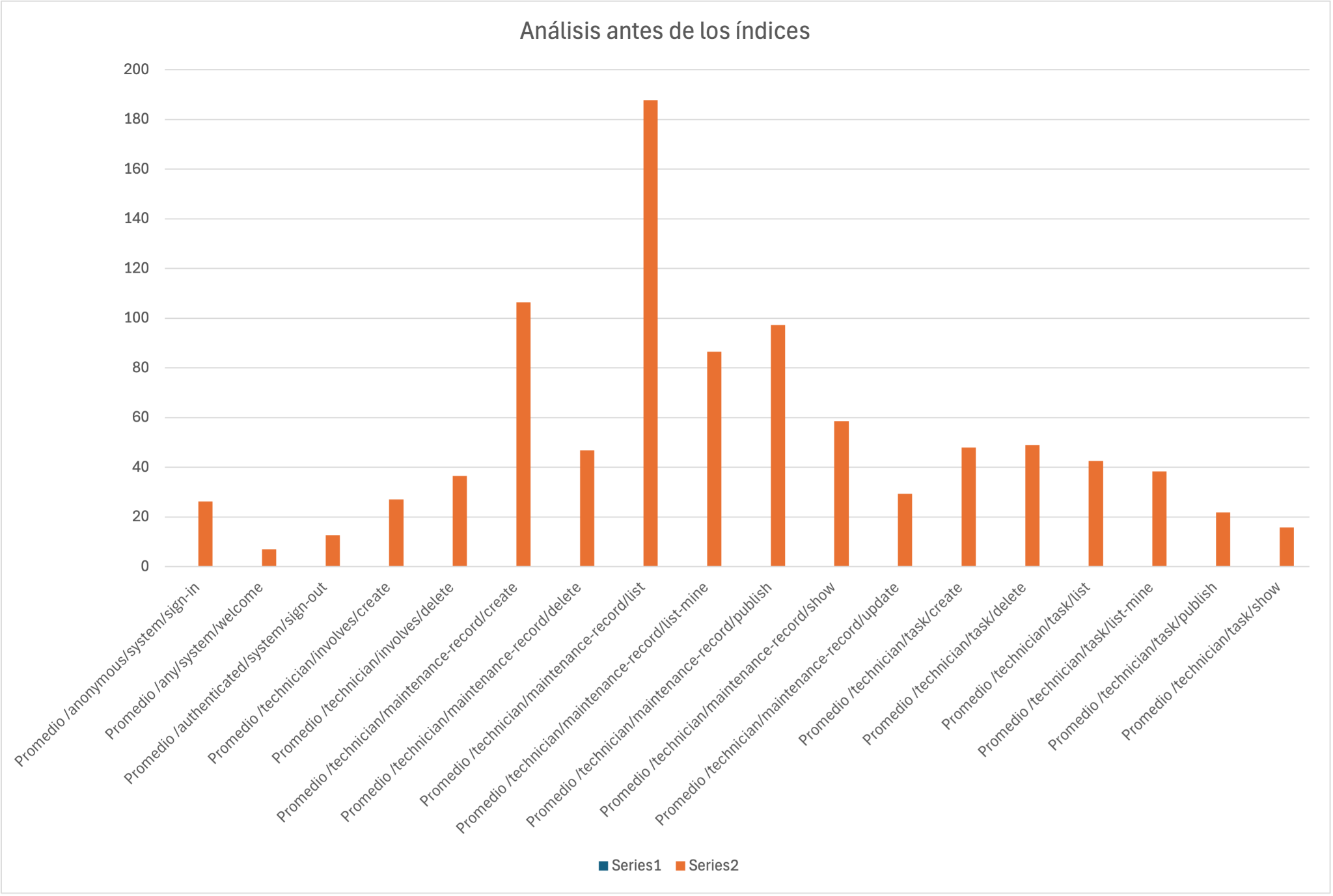
| **.safe** | **Description** | **Bugs detected** |
| --- | --- | --- |
| list | - The technicians’ records were listed, including their own records (which may be unpublished) as well as the catalog of records published by all technicians. | No bugs were found. |
| show | - Published and unpublished maintenance records of the logged-in technician were displayed. - Published records from other technicians were also shown. | No bugs were found. |
| update | For unpublished records of the logged-in technician: - An attempt was made to submit an empty form. - Each variation of every field in the maintenance record creation form was tested. | No bugs were found. |
| publish | For unpublished records of the logged-in technician: - An attempt was made to submit an empty form. - Each variation of every field in the maintenance record publication form was tested, including attempts to publish records with both published and unpublished tasks. | No bugs were found. |
| delete | For unpublished records of the logged-in technician: - Attempts were made to delete maintenance records with tasks linked to them (both published and unpublished tasks), as well as records without any linked tasks. | No bugs were found. |
| **.hack** | **Description** | **Bugs detected** |
| list | - Attempts were made to list maintenance records without a role or with an incorrect role (both personal records and the full catalog). | No bugs were found.. |
| show | **Incorrect role:** - An attempt was made to make the request without an id parameter - Attempts were made to display tasks with ids that do not exist in the database - Attempts were made to access the maintenance records view screen, both published and unpublished  **With technician role:** - An attempt was made to make the request without an id parameter - Attempts were made to display tasks with ids that do not exist in the database - Attempts were made to display unpublished records NOT assigned to the logged-in technician | No bugs were found. |
| create | **Incorrect role:** - Attempts were made to access the task creation screen.  **With technician role:** - Attempts were made to hack the id by using values that do not exist in the database, including other published and unpublished records of the logged-in technician and those of another technician. - Attempts were made to assign invalid values to the status field, such as 9999, null, or strings not included in the enum values. - Attempts were made to hack the aircraft field by assigning invalid values like null or non-existent ids. | It was necessary to modify the authorise method to adjust the validation of the TaskType enum, and unreachable code was removed. |
| update | **Incorrect role:**  - An attempt was made to make the request without an id parameter - An attempt was made to make the request with an id that does not exist in the database - Attempts were made to update both published and unpublished tasks  **With technician role:** - An attempt was made to make the request without an id parameter - Attempts were made to update records with non-existent ids - Attempts were made to update unpublished records of another technician - Attempts were made to update published records of the logged-in technician - Attempts were made to update published records of another technician - Attempts were made to assign invalid values to the Status field, such as 9999, null, or strings not included in the enum values - Attempts were made to hack the aircraft field by assigning invalid values like null or non-existent ids | It was necessary to modify the authorise method to adjust the validation of the TaskType enum, and unreachable code was removed. |
| publish | **Incorrect role:** - An attempt was made to make the request without an id parameter - An attempt was made to make the request with an id that does not exist in the database - Attempts were made to publish both published and unpublished records  **With technician role:** - An attempt was made to make the request without an id parameter. - Attempts were made to publish records with non-existent ids - Attempts were made to publish unpublished records of another technician - Attempts were made to publish published records of the logged-in technician - Attempts were made to publish published records of another technician - Attempts were made to assign invalid values to the Status field, such as 9999, null, or strings not included in the enum values - Attempts were made to hack the aircraft field by assigning invalid values like null or non-existent ids | It was necessary to modify the authorise method to adjust the validation of the TaskType enum, and unreachable code was removed. |
| delete | **Incorrect role:** - An attempt was made to make the request without an id parameter - Attempts were made to delete published, unpublished, and non-existent records  **With technician role:** - Attempts were made to delete records with non-existent ids - Attempts were made to delete existing records, both published and unpublished, of another technician - Attempts were made to delete existing published records of the logged-in technician | No bugs were found. |

**Involves**

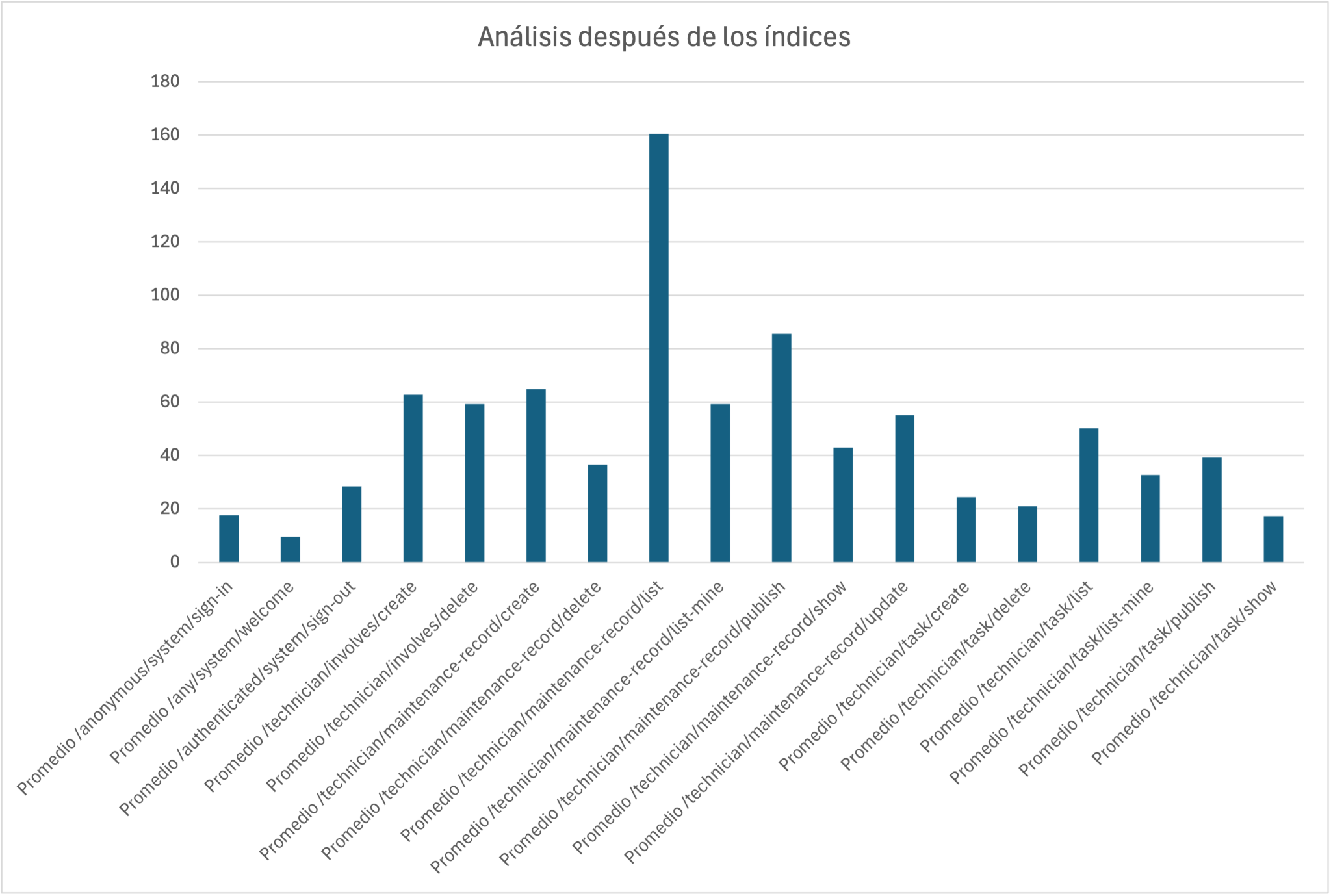
| **.safe** | **Description** | **Detected bugs** |
| --- | --- | --- |
| create | - An attempt was made to link tasks available in the dropdown menu, including testing the option with no task selected (“---”). | No bugs were found. |
| delete | - An attempt was made to unlink tasks available in the dropdown menu, including testing the option with no task selected (“---”). | No bugs were found. |
| **.hack** | **Description** | **Detected bugs** |
| create | **Incorrect role:** - Attempts were made to access the task-linking screen for maintenance records using invalid ids like 9999 (which do not exist in the database), as well as existing records (both published and unpublished).  **With technician role:**  - Access was attempted to the linking form of another technician’s maintenance record (both published and unpublished) or with a non-existent id like 9999. - Access was attempted to the linking form of a published record owned by the logged-in technician. - Attempts were made to link a task with a non-existent id like 9999 to a maintenance record. - Attempts were made to link an existing but unpublished task. (NOTE: In D04 follow up, my lab instructor warned me that a Maintenance Record—whether published or unpublished—cannot have unpublished tasks associated. This is because tasks don’t make sense if they’re incomplete or could be deleted. I mention this due to some confusion.) - Attempts were made to link a task that was already linked to that record. | No bugs were found.  . |
| delete | **Incorrect role:** - Attempts were made to access the task unlinking screen for maintenance records using invalid ids like 9999, which do not exist in the database, as well as existing records (both published and unpublished).  **With technician role:** - Access was attempted to the unlinking form of another technician’s maintenance record (both published and unpublished) or with a non-existent id like 9999. - Access was attempted to the unlinking form of a published record owned by the logged-in technician. - Attempts were made to unlink a task with a non-existent id like 9999 from a maintenance record. - Attempts were made to unlink an existing but unpublished task. - Attempts were made to unlink a task that was not linked to that record. | No bugs were found. |

## 4.2 Performance testing

Performance testing was carried out on the same machine. To do this, all the previously mentioned traces were executed twice: first without database indexes, and then again with indexes added, aiming to enhance performance by reducing query response times. The timing data from both runs are presented below:



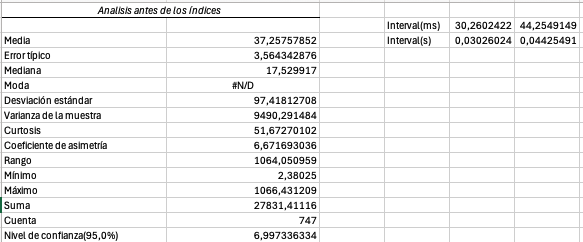
Average times after indexing

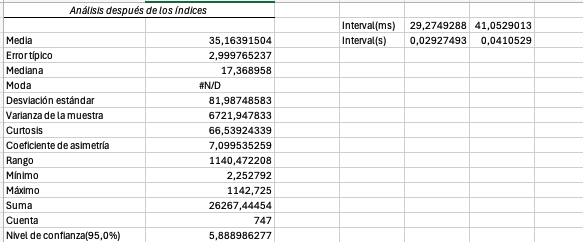


Average times after indexing

The graphs show that using database indexes helps improve performance by reducing the time needed to resolve queries. Indexes allow the database to locate relevant data more efficiently without scanning entire tables, which leads to faster response times. For example, the average of /technician/maintenance-record/list before indexes is rounding 190ms and after indexes is 160ms.

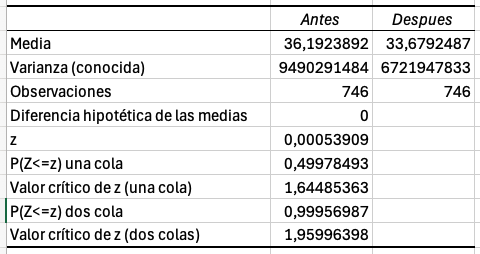
To provide a thorough comparison of the application’s performance before and after applying the indexes, the relevant statistics from each analysis are presented





Looking at the confidence intervals, before applying the index changes, we could guarantee responding to any request within 44.2549 ms or less with 95% confidence. After optimizing the database queries through indexing, this upper time limit decreased to 41.0592 ms, a reduction of 3.1957 ms. This represents an approximate 7.22% improvement in application performance.

\*I have observed that, since I conducted the tests using a Virtual Machine, the response times I recorded are noticeably higher compared to those of my classmates.



Despite the reduction in average times, the obtained p-value (0.9957), which falls within the interval (α, 1.00], indicates that the use of indexes did not lead to a statistically significant improvement.

# Conclusions

This testing report has covered both functional and performance testing of the Acme-ANS project. The functional tests demonstrated that all key features work as expected, with comprehensive coverage of positive, negative, and security-related test cases. No significant bugs were found during these tests, indicating a stable and reliable application behavior under various scenarios.

Performance testing focused on evaluating the impact of database indexing on query response times. While the results showed some improvement in average response times—reducing the delay by approximately 7%—the statistical analysis revealed that this improvement was not significant under the tested conditions. This suggests that while indexing can contribute to performance enhancements, other factors may also play a role in overall system responsiveness.

Future work could include profiling specific bottlenecks using software profiling tools and exploring other optimization strategies, such as query refactoring or hardware improvements, to further enhance performance. Additionally, testing under varied and more intensive workload conditions could provide a clearer picture of the system’s behavior in real-world usage.

Overall, this report confirms that the Acme-ANS project functions correctly and performs reasonably well, with potential for further improvements as development continues.

# Bibliography

Intentionally blank.