|  |
| --- |
| José Coronil Álvarez (joscoralv@alum.us.es)  Miguel García Vizcaíno (miggarviz@alum.us.es)  Javier Ignacio Milá de la Roca Dos Santos (javmildos@alum.us.es)  Emilio Manuel Vázquez Cruz (emivazcru@alum.us.es)  Group C1.028  5-16-2025 |

|  |
| --- |
| Acme AirNav Solutions, Inc. |
| **Testing Report** |
| https://github.com/Emilio-115/DP2-Acme-ANS |



Contents

[Executive Summary 2](#_Toc190960965)

[Revision Table 2](#_Toc190960966)

[Introduction 3](#_Toc190960967)

[Content 3](#_Toc190960968)

[Conclusions 5](#_Toc190960969)

[Bibliography 5](#_Toc190960970)

# Executive Summary

This report documents the tests implemented for Student 3’s tasks. All required test cases were created and executed, covering both normal and unauthorized scenarios. Most features functioned as expected, and the few bugs identified were fixed. All reachable lines of code were covered by the tests. Performance was analyzed before and after adding indices, showing a slight slowdown.

# Revision Table

|  |  |  |
| --- | --- | --- |
| Revision number | Date | Description |
| 1 | 16/05/2025 | Initial version |
| 2 | 24/05/2025 | First completed version |

# Introduction

This document is the testing report for Student 3’s deliverables in the Acme-ANS project. It includes a summary of the test cases developed for the features implemented, covering both expected use cases and security checks against unauthorized access. It also reports on the achieved test coverage and a basic performance comparison before and after database indexing.

# Content

In accordance with the requirements, both *safe* and *hack* tests were recorded for all flight assignment and activity log features until full coverage was accomplished, and performance was analyzed both before and after adding indices

## Tests Developed

The following is a list of the test cases implemented, grouped by feature:

### Flight Assignments

* **list-drafts.safe**, **list-drafts.hack**,  
  **list-planned.safe**, **list-planned.hack**,  
  **list-departed.safe**, **list-departed.hack**:  
  These test the list of assignments that are unpublished, of flights that haven’t taken off and of flights that have taken off, respectively. The hacked versions tried to access the list as a user without the right role. They found no bugs.
* **show.safe**:  
  This tests showing the data. The safe case guarantees you can see the data for all three types of flights with their appropriate options (e.g. cancelling and confirming for pending assignments of pending flights, updating and publishing unpublished assignments) (though it should be noted that the presence of UI buttons is not strictly verified by the test). It found that unavailable members could still see buttons for editing, which would result in an authorization failure.
* **show.hack**:  
  This tests trying to access the data without the right role and as the wrong user, for all three types of data. It found no bugs.
* **create.safe**, **update.safe**, **delete.safe**, **publish.safe**, **cancel.safe**, **confirm.safe**:  
  These test that you can create, update, delete, publish, cancel and confirm an assignment, and the necessary validators. They found no bugs.
* **create.hack**:  
  This tests that you can’t create without having the right role, and that you can’t alter the id or navigation attributes. It found that altering the id wouldn’t necessarily get rejected, which was fixed.
* **update.hack**, **delete.hack**, **publish.hack**:  
  These tests that you can’t update, delete or publish without having the right role, nor other user’s assignments, nor your published assignments, and that you can’t alter the navigation attributes. They found no bugs.
* **confirm.hack**, **cancel.hack**:  
  These test that you can’t confirm or cancel without having the right role, nor other user’s assignments, nor your unpublished assignments, nor your assignments of legs that have already departed, and that you can’t alter read-only attributes. They found no bugs.
* **unavailable-member.hack**:  
  This tests that you can’t create, update, delete, publish, cancel or confirm if you’re not logged in as a member who is available. It found no bugs.

### Activity Logs

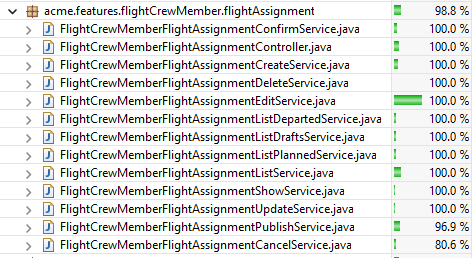
* **list.safe**, **list.hack**,  
  **list-drafts.safe**, **list-drafts.hack**   
  These test the list of published and unpublished activity logs, respectively. The hacked versions tried to access the list as a user without the right role. Also, it tests that you can filter published activity logs by a published, confirmed assignment of a flight that departed that is yours, and that it fails otherwise. They found that you could try to filter by any flight assignment, and that though it wouldn’t work with invalid ones or ones belonging to other users, it wouldn’t fail. This was fixed
* **show.safe**:  
  This tests showing the data. The safe case guarantees you can see the data for published and unpublished assignments, and that it would only show options to modify for non-drafts. It found no bugs
* **show.hack**:  
  This tests trying to access the data without the right role and as the wrong user, for both published and unpublished activity logs. It found no bugs.
* **create.safe**, **update.safe**, **delete.safe**, **publish.safe**:  
  These test that you can create, update, delete and publish an activity log, and the necessary validators. They found no bugs.
* **create.hack**:  
  This tests that you can’t create without having the right role, and that you can’t alter the id or navigation attributes. It found that altering the id wouldn’t necessarily get rejected, which was fixed.
* **update.hack**, **delete.hack**, **publish.hack**:  
  These tests that you can’t update, delete or publish without having the right role, nor other user’s assignments, nor your published assignments, and that you can’t alter the navigation attributes. They found no bugs.

## Test Coverage

The following is a summary of the coverage achieved for all relevant files

### Flight Assignments

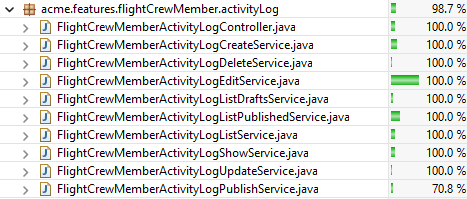
* **Validator** (93.8%):  
  The missing coverage is the result of validation paths being inaccessible through user operations since they’re handled in the *authorize* method first, and validation paths never being skipped even though they theoretically could be due to courtesy null checks (e.g. checking that the leg of a FlightAssignment is not null before accessing its properties to prevent exceptions despite legs being mandatory).
* **ConfirmService**, **Controller**, **CreateService**, **DeleteService**, **EditService**, **ListDepartedService**, **ListDraftsService**, **ListPlannedService**, **ListService**, **ShowService**, **UpdateService** (100%):  
  These classes were entirely covered by the tests.
* **PublishService** (96.9%):  
  The missing coverage is the result of authorization paths never being skipped even though they theoretically could be due to null checks.
* **CancelService** (80.6%):  
  The missing coverage is due to the unbind function not being covered, since any flight assignment authorized for cancelling can be cancelled.





### Activity Logs

* **Validator** (84.9%):  
  The missing coverage is the result of validation paths being inaccessible through user operations since they’re handled in the *authorize* method first and validation paths never being skipped even though they theoretically could be due to null checks.
* **Controller**, **CreateService**, **DeleteService**, **EditService**, **ListDraftsService**, **ListPublishedService**, **ListService**, **ShowService**, **UpdateService** (100%):  
  These classes were entirely covered by the tests.
* **PublishService** (70.8%):  
  The unbind function is not covered. Since any activity log that is valid as a draft is also valid when published, it is impossible to fail validation. Thus, the unbind method is inaccessible.





## Performance Analysis

The performance of the project was analyzed according to the specifications both before and after indices were created for the relevant tables.

The times were recorded executing only the tests regarding flight crew member features. The exact same tests were run with both; the only difference is the indexing of the tables. The specific results can be seen on the Excel files in this folder.

### Before Indices

Before the indices, the testing resulted in a 95% confidence interval from 8.99ms to 10.36ms and a mean of 9.58ms. The costliest operation was cancelling a flight assignment, which in average took 33.39ms.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Response time* | |  |  |  |  |
|  |  |  | *Interval (ms)* | 8.991419 | 10.3647 |
| Mean | 9.678059 |  | *Interval (s)* | 0.008991 | 0.010365 |
| Standard Error | 0.349847 |  |  |  |  |
| Median | 6.9195 |  |  |  |  |
| Mode | 2.1847 |  |  |  |  |
| Standard Deviation | 10.33677 |  |  |  |  |
| Sample Variance | 106.8488 |  |  |  |  |
| Kurtosis | 16.87884 |  |  |  |  |
| Skewness | 3.076939 |  |  |  |  |
| Range | 104.3643 |  |  |  |  |
| Minimum | 1.563 |  |  |  |  |
| Maximum | 105.9273 |  |  |  |  |
| Sum | 8448.946 |  |  |  |  |
| Count | 873 |  |  |  |  |
| Confidence Level (95.0%) | 0.68664 |  |  |  |  |

### After indices

After the indices, the interval was from 10.25ms to 11.95ms, with a mean of 10.99ms. The costliest operation was confirming a flight assignment, which in average took 36.72ms. Cancelling a flight assignment was a close second, taking 36.21ms in average.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Response time* | |  |  |  |  |
|  |  |  | *Interval (ms)* | 10.24744 | 11.95314 |
| Mean | 11.10029 |  | *Interval (s)* | 0.010247 | 0.011953 |
| Standard Error | 0.434532 |  |  |  |  |
| Median | 7.8506 |  |  |  |  |
| Mode | 2.3173 |  |  |  |  |
| Standard Deviation | 12.83892 |  |  |  |  |
| Sample Variance | 164.8379 |  |  |  |  |
| Kurtosis | 24.11654 |  |  |  |  |
| Skewness | 3.767839 |  |  |  |  |
| Range | 134.898 |  |  |  |  |
| Minimum | 1.5918 |  |  |  |  |
| Maximum | 136.4898 |  |  |  |  |
| Sum | 9690.553 |  |  |  |  |
| Count | 873 |  |  |  |  |
| Confidence Level (95.0%) | 0.85285 |  |  |  |  |

### Comparison

The two sample z-test for means resulted in a P(Z<=z) two-tail value of 0.0121, which is below significantly below the alpha value for the 95% confidence level (0.05). Thus, the means can be compared with confidence. And, since the mean response time is higher for the tests performed with indices, it can be concluded that adding the indices decreased performance.

This might be because of the small scale of the tests and the sample database resulting in the overhead cost of indices not being offset by the performance gains on querying the database.

|  |  |  |
| --- | --- | --- |
| z-Test: Two Sample for Means |  |  |
|  |  |  |
|  | *Before indices* | *After indices* |
| Mean | 9.588754702 | 10.98820539 |
| Known Variance | 106.84878 | 164.8379173 |
| Observations | 872 | 872 |
| Hypothesized Mean Difference | 0 |  |
| z | -2.507157719 |  |
| P(Z<=z) one-tail | 0.00608532 |  |
| z Critical one-tail | 1.644853627 |  |
| P(Z<=z) two-tail | 0.012170639 |  |
| z Critical two-tail | 1.959963985 |  |

# Conclusions

All required tests were written and executed successfully. The features behaved correctly in both valid and invalid scenarios. The tests helped identify bugs, which were subsequently fixed. Test coverage was achieved for all reachable lines in validators and services. Performance tests showed a small slowdown after indexing, likely due to the small database size not benefiting from optimization.

# Bibliography

Intentionally blank.