Verification and Validation Report: Software Engineering

Team 1, BANDwidth
Declan Young
Ben Dubois
Nathan Uy
Aidan Mariglia

April 4, 2025

1 Revision History

Date	**Version**	**Notes**
March 7, 2025	1.0	VnV results
March 18, 2025	2.0	Addressed peer review

2 Symbols, Abbreviations and Acronyms

symbol	**description**
T	Test
M	Module
FR	Functional Requirements
NFR	Non-functional Requirements

Contents

List of Tables

List of Figures

This Verification and Validation report evaluates the system's functional and non-functional requirements. It also documents the unit tests and the changes made due to testing and feedback. Finally, it outlines the traces to requirements and modules, to ensure the system meets its objectives.

3 Functional Requirements Evaluation

3.1 Algorithm Submission

1. test-FR1-ST1:

   Initial State: A logged in user on the algorithm submission page

   Input: A valid .mat submission

   Expected Output: Submission appears as 'pending' on submission page

   Actual Output: Submission appears as 'pending' on submission page. Submission in progress

   Result: Pass

2. test-FR1-ST3:

   Initial State: A logged in user on the algorithm submission page

   Input: An algorithm that exceeds the maximum file size

   Expected Output: Error indicating the submission exceeds the max file size

   Actual Output: Submission page indicates that the file was submitted successfully

   Result: Fail

3. test-FR1-ST4:

   Initial State: A user is logged-in and on the submit page.

   Input: A .txt file.

   Expected Output: An error message indicating an invalid file was submitted.

   Actual Output: An error message indicating an invalid file was submitted. File Not Supported Error

   Result: Pass

3.2 Results Reporting

1. test-FR2-ST1:

   Initial State: A logged in user on the algorithm submission page

   Input: Submission page indicates that the submission has completed

   Expected Output: Submission page indicates that the submission has completed. All of the submission statistics are updated.

   Actual Output: Submission page indicates that the submission has completed. All of the submission statistics are updated. Submission completed

   Result: Pass

2. test-FR2-ST2:

   Initial State: A logged in user on the algorithm submission page

   Input: An invalid algorithm

Expected Output: Submission page indicates that the submission has failed. None of the submission statistics are updated.

Actual Output: Submission page indicates that the submission has failed. None of the submission statistics are updated. Submission failed

Result: Pass

3.3 Parallel Execution

1. test-FR3-ST1:

   Initial State: System is executing 1 model with a long runtime

   Input: A model with a short runtime

   Expected Output: The model submitted second with a short runtime should finish execution before the long running model submitted first

   Actual Output: The second model finishes executing second, and takes significantly longer than it does when it is the only model executing. Parallel execution failed

   Result: Fail

3.4 Account Creation

1. test-FR4-ST1:

   Initial State: User is on the registration page

   Input: unique username and email, password, first and last name, and academic affiliation.

   Expected Output: Redirection to the Intro page.

   Actual Output: Redirection to the Intro page.

   Result: Pass

2. test-FR4-ST2:

   Initial State: User is on the registration page.

   Input: a username or email that is already in the database, any password, first and last name, and academic affiliation.

   Expected Output: Account already exists error.

   Actual Output: Account already exists error. Error Message

   Result: Pass

3.5 User Login

1. test-FR5-ST1:

   Initial State: User is on the login page.

   Input: A username and password pair that exists in the database.

   Expected Output: Redirection to the home page.

   Actual Output: Redirection to the home page with a welcome message for the user logged-in. Error Message

   Result: Pass

2. test-FR5-ST2:

   Initial State: User is on the login page.

   Input: A username and password pair that does not exist in the database.

   Expected Output: An error message indicating invalid credentials.

   Actual Output: An error message saying username and password didn't match. Error Message

   Result: Pass

3.6 Data Segregation

1. test-FR6-ST1:

   Initial State: User is logged in

   Input: Id of a submission to retrieve

   Expected Output: The submission details corresponding to the id

   Actual Output: The submission details corresponding to the id Submission retrieval

   Result: Pass

2. test-FR6-ST2:

   Initial State: A user is logged-in.

   Input: The user tries to access a submission from a different user.

   Expected Output: 404 error.

   Actual Output: 404 error. 404 Error Message

   Result: Pass

3.7 Leaderboard Access

1. test-FR8-ST1:

   Initial State: User is logged in

   Input: Navigation to leaderboard page

   Expected Output: Leaderboard is visible, contianing all past mubmissions from all users

   Actual Output: Leaderboard is visible, contianing all past mubmissions from all users Leaderboard

   Result: Pass

3.8 Leaderboard Categorization

1. test-FR9-ST1:

   Initial State: The user is logged-in and on the leaderboard page.

   Input: Filter the leaderboard on category model_type="Kalman Filter"

   Expected Output: The entries of the leaderboard only contains model submissions with the Kalman Filter model type.

   Actual Output: The leaderboard entries did not change after applying the filter.

   Result: Fail

3.9 Sort Leaderboards

1. test-FR10-ST1:

   Initial State: An unsorted leaderboard with 10 entries. Unsorted leaderboard

   Input: Sort by the weighted error column in descending order.

   Expected Output: A sorted leaderboard table in descending order based on the weighted error.

   Actual Output: A sorted leaderboard table in descending order based on the weighted error. Sorted leaderboard

   Result: Pass

3.10 Execution Progress

1. test-FR11-ST1:

   Initial State: A logged in user on the algorithm submission page

   Input: A valid algorithm

   Expected Output: Submission begins in 'pending' state, before eventually reaching a 'completed' state.

   Actual Output: Submission begins in 'pending' state, before eventually reaching a 'completed' state. Submission in progress; Submission completed

   Result: Pass

4 Nonfunctional Requirements Evaluation

4.1 Look and Feel

1. test-NFR1-ST1

   Initial State: The website is open in a web browser

   Input: User navigates the whole SOCAlgoTestPlatform website and the whole Kaggle website

   Output: User information on any similarities and differences that the user notices between websites

   Result: After the user had navigated through both websites, they noted that both websites had similar login and registration processes as well as similar headers at the top of the screen that navigated to core functionalities. However, the user also noted that Kaggle had a search feature that the SOCAlgotTestPlatform does not yet currently have.

4.2 Learning

1. test-NFR2-ST1

   Initial State: User is on the intro page of the website and are brand new to the system

   Input: User navigates to the walkthrough video and watches this video in full

   Output: Walkthrough video successfully plays and the user understands basic functionality

   Result: Video successfully played and after the user had finished watching they were able to successfully perform basic functionalities such as submitting an algorithm and viewing the leaderboard.

4.3 Usability

1. test-NFR3-ST1

   Initial State: The website is open in a web browser

   Input: The user is asked to navigate the entire SOCAlgoTestPlatform website

   Output: A list created by the user of all features or symbols that are unfamiliar to the user or ambiguous

   Result: The user navigated the entire system and could not identify any symbols or features that they found to be ambiguous or unfamiliar.

2. test-NFR3-ST2

   Initial State: The website is open in a web browser

   Input: The user is asked to skim through all terminology used within the website

   Output: A list created by the user of all inconsistencies between terminology

   Result: The user skimmed through all terminology used on the homepage, the submission screens, and the leaderboard and was unable to find any inconsistencies.

3. test-NFR3-ST3

   Initial State: The user is logged into the system

   Input: The user goes through all steps to submit an algorithm

   Output: The user receives a message after submission that either indicates success or failure

   Result: The user first submitted an algorithm through the submissions page that was not a Matlab file and they received a failure message indicating the file needs to be .m or .mat. The user then submitted a Matlab file and they received a pending message indicating the submission was successful.

4. test-NFR3-ST4

   Initial State: The user is logged into the system

   Input: The user is asked to submit an algorithm to the system with no assistance

   Output: The user successfully submits the algorithm with only the help from documentation

   Result: When asked to submit an algorithm, the user first tried to navigate to the submission page however, once they arrived at this page they were unsure what was needed to submit. The user then navigated back to the home page and found the walkthrough video. They proceeded to watch the section in this video on submitting an algorithm. After this, the user was able to successfully submit an algorithm that already existed on the testing computer.

4.4 Speed and Latency

1. test-NFR4-ST1

   Initial State: User is logged in on home page, leaderboard contains entries

   Input: User selects leaderboard tab

   Output: The leaderboard is displayed nearly instantaneously, well below the 3 second threshold (using realistic latency situations) defined in the VnVPlan

   Result: Pass

4.5 Authentication

1. test-NFR5-ST1

   Refer to test-FR5-ST1 and test-FR5-ST2

4.6 Database Integrity

1. test-NFR6-ST1

   Refer to Unit Tests for Models. Test case: test_invalid_submission_creation()

4.7 Robustness or Fault Tolerance

1. test-NFR7-ST1

   Initial State: Running system, authenticated user

   Input: Invalid model which contains syntax error

   Output: Model execution fails, second user is still able to submit models and use the system as normal

   Result: This test was successful, the system behaved as described in the test case, and the error was isolated.

4.8 Adaptability

1. test-NFR10-ST1

   Initial State: Running system accessed through Firefox and Chrome

   Input: Set of tests defined in VnVPlan

   Output: Test results unchanged across both browsers. Chrome Firefox Microsoft Edge

   Result: Pass

5 Unit Testing

5.1 Database Integrity

Unit Tests for Models

5.2 File Submission

Unit Tests for Forms

5.3 Data Segregation

Unit Tests for Views

5.4 Leaderboard Sorting

Unit Tests for Leaderboard

6 Changes Due to Testing

6.1 Change due to supervisor's feedback

During our Rev0 demo with our supervisor, Dr. Phil, he gave our team feedback on several features. Firstly, he wanted the leaderboard to be available to the public, not just logged-in users. Secondly, he pointed out that upon submission, we are supposed to ask users for the type of model they are submitting such as Machine Learning or Kalman Filter models, so that this field could be used for filtering the results. Thirdly, he wanted to add a link to another web page where users could download the data to be used for creating the model. Lastly, we initially set up the system so that after a user registers for an account, Dr. Phil's approval as an admin would be needed before the user could use the system. However, Dr. Phil suggested to scrap this feature and instead just do an email verification upon registration.

6.2 Change due to survey feedback

From the survey feedback, we were able to highlight areas for improvement in the SOCAlgoTestPlatform. Users noted the absence of a search feature, which will be considered for future implementation but is not a current priority. Submission instructions will be clarified to reduce confusion about required file formats and size. To accommodate different learning preferences, step-by-step text guides and FAQs will supplement the walkthrough video. Additionally, minor inconsistencies in error messages will be reviewed for clarity. These changes will improve usability, accessibility, and overall user experience based on direct user input.

6.3 Change due to failing test case

The test for an algorithm that exceeds the maximum file size failed as our system still accepted large file sizes. As a result, we will add a functionality to limit the file size accepted and display an error message to the user. In addition, the test for parallel execution failed, so changes to the model execution module were implemented to allow parallel processing of algorithm submissions. Lastly, the filter for model type was not working properly, so changes were made so that leaderboard entries could be filtered by model type correctly.

7 Automated Testing

Our team used Django's built-in test framework, which was built on Python's unittest module. It allowed us to test different django components such as models, forms, and views. Test cases are written as classes that inherit TestCase, and assert statements are used to compare the expected and actual outputs. The use of the set-up method also allowed for the reuse of data and automatic set up. Django's client allows for mocking HTTP requests.

8 Trace to Requirements

Test ID	R1	R2	R3	R4	R5	R6	R8	R9	R10	R11
test-FR1-1	×									
test-FR1-2	×									
test-FR1-3	×									
test-FR2-1		×								
test-FR2-2		×								
test-FR3-1			×							
test-FR4-1				×						
test-FR4-2				×						
test-FR5-1					×					
test-FR5-2					×					
test-FR6-1						×				
test-FR6-2						×				
test-FR8-1							×			
test-FR9-1								×		
test-FR10-1									×	
test-FR11-1										×

Table 1: **Functional Requirements Traceability**

Test ID	LFR1	SR1	LR1	LR2	UPR1	UPR2	UPR3	UPR4	ADR1
test-LFR-1	×	×							
test-LR-1			×						
test-UPR-1					×				
test-UPR-2						×			
test-UPR-3							×		
test-UPR-4								×	
test-ADR-1									×

Table 2: **Non-Functional Requirements Traceability Part 1**

Test ID	SLR1	SLR2	SLR3	ACR1	IR1	RR1
test-SLR-1	×	×	×			
test-ACR-1				×		
test-IR-1					×	
test-RR-1						×

Table 3: **Non-Functional Requirements Traceability Part 2**

9 Trace to Modules

Test ID	M3	M4	M5	M6	M8	M9	M10	M11	M12	M13	M14	M15	M16
test-FR1-1										×			
test-FR1-2										×			
test-FR1-3										×			
test-FR2-1													×
test-FR2-2													×
test-FR3-1							×	×	×				
test-FR4-1			×										
test-FR4-2			×										
test-FR5-1		×											
test-FR5-2		×											
test-FR6-1				×									
test-FR6-2				×									
test-FR8-1												×	
test-FR9-1												×	
test-FR10-1												×	
test-FR11-1										×			

Table 4: **Implemented Modules Traceability Part 1**

Test ID	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15	M16
test-LFR-1		×	×			×	×				×	×	×	×
test-LR-1						×	×							
test-UPR-1		×	×			×	×				×	×	×	×
test-UPR-2		×	×			×	×				×	×	×	×
test-UPR-3		×	×			×	×				×	×	×	×
test-UPR-4		×	×			×	×				×	×	×	×
test-ADR-1	×	×	×	×	×	×	×	×	×	×	×	×	×	×
test-SLR-1												×	×	
test-ACR-1		×												
test-IR-1				×										
test-RR-1								×						

Table 5: **Implemented Modules Traceability Part 2**

References

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Reflection.

The purpose of reflection questions is to give you a chance to assess your own learning and that of your group as a whole, and to find ways to improve in the future. Reflection is an important part of the learning process. Reflection is also an essential component of a successful software development process.

Reflections are most interesting and useful when they're honest, even if the stories they tell are imperfect. You will be marked based on your depth of thought and analysis, and not based on the content of the reflections themselves. Thus, for full marks we encourage you to answer openly and honestly and to avoid simply writing "what you think the evaluator wants to hear."

Please answer the following questions. Some questions can be answered on the team level, but where appropriate, each team member should write their own response:

1. What went well while writing this deliverable?
2. What pain points did you experience during this deliverable, and how did you resolve them?
3. Which parts of this document stemmed from speaking to your client(s) or a proxy (e.g. your peers)? Which ones were not, and why?
4. In what ways was the Verification and Validation (VnV) Plan different from the activities that were actually conducted for VnV? If there were differences, what changes required the modification in the plan? Why did these changes occur? Would you be able to anticipate these changes in future projects? If there weren't any differences, how was your team able to clearly predict a feasible amount of effort and the right tasks needed to build the evidence that demonstrates the required quality? (It is expected that most teams will have had to deviate from their original VnV Plan.)

Nathan Uy

1. What went well while writing this deliverable was that we did not have to change much of our VnVPlan so it was easy to just test out every scenario and report the output.
2. Testing the non-functional requirements was difficult as it is not as objective as the functional requirements where there is one right answer. But, by creating surveys and metrics we were able to measure how well our system performed according to the test plan.

Declan Young

1. During this deliverable running the manual tests went well. This is because we had a good idea of what needed to be tested for our application, and how to do the tests manually, so it was just a matter of executing these tests and documenting the results.
2. One pain point we encountered during this deliverable was documenting particular tests that are difficult to show the validation of For example, for a parallel execution, it was difficult o show manually that this feature was actually working in our application. It may have been better to use automated tests for to validate particular features like this to make it more clear how the validation was being done and to have more confidence in the results.

Aidan Mariglia

1. One thing that went well during this deliverable was the system performance. The majority of our tests passed, and the ones which did not were predictable, and did not come as a surprise. This increases confidence that the system is well implemented and matches our mental model.

2. One pain point was closing the gap between the VnV plan and the current goals of the system. Time has passed, more of the system has been implemented, and more feedback has been received. As such some of the tests in the VnV plan are no longer applicable, and needed to be modified.

Benjamin Dubois

1. One thing that went well during this deliverable were the usability tests. It can be hard to predict how it will go when you have a completely new user using the system as up to this point the only people that have used it have been developers, but during this testing, the users were able to learn the system very quickly and struggled to find any confusing aspects.

2. One pain point we encountered during this deliverable was sorting through which of our tests from our VnV plan were still viable and which were no longer needed. We have made a lot of changes to the system since we wrote our VnV plan, so there was a significant amount of tests in this VnV that we ended up getting rid of because they were no longer necessary after these changes. This caused a pain point as before each test we had to analyze if the test was still necessary based on the current state of the system.

Team

3. Section 6 (changes due to testing) primarily came from speaking to Dr. Kollmeyer as well as other proxies that did usability tests on our application. This is because although we, as a group, had a vision for the application, we thought it was very important to get validation, feedback and criticism from other sources, so that we could make the necessary changes to fix or improve the application.

   The other sections in this document, for the most part, did not come from other sources. This is because most of the other sections for example, evaluation of functional requirements, were just a product of executing what we had already outlined in the verification and validation plan we outlined. As a result of this, we tried to follow this plan as closely as possible to have structure when testing, and having confidence in the results of out verification and validation.

4. One of the major changes to the verification and validation plan we originally outlines was not having as much automated testing as originally anticipated. This required changing the plan to include more manual testing to replace these automated tests, to ensure that all features wer still properly tested. These changes occurred mostly due to time constraints, as the time and effort required to created many of the automated tests originally planned would have taken too much time and prevented thorough validation and verification of the entire application. For future projects, we anticipate these changes by having better estimates for the time different verification and validation tasks will take. This will allow us to budget time better, and ensure all tasks are accounted for properly. Additionally, in the future we can try to allocate more time to validation and verification, so less changes to the plan need to be made due to time constraints.