Project Title: System Verification and Validation Plan for Software Engineering

Team #2, Campus Connections
Waseef Nayeem
Zihao Du
Matthew Miller
Firas Elayan
Abhiram Neelamraju
Michael Kim

October 30, 2023

Revision History

Date	Version	Notes
Oct 29	1.0	Add Functional Requirements Tests
Date 2	1.1	Notes

[The intention of the VnV plan is to increase confidence in the software. However, this does not mean listing every verification and validation technique that has ever been devised. The VnV plan should also be a **feasible** plan. Execution of the plan should be possible with the time and team available. If the full plan cannot be completed during the time available, it can either be modified to "fake it", or a better solution is to add a section describing what work has been completed and what work is still planned for the future. —SS]

[The VnV plan is typically started after the requirements stage, but before the design stage. This means that the sections related to unit testing cannot initially be completed. The sections will be filled in after the design stage is complete. the final version of the VnV plan should have all sections filled in. —SS]

Contents

List of Tables

[Remove this section if it isn't needed —SS]

List of Figures

[Remove this section if it isn't needed —SS]

1 Symbols, Abbreviations, and Acronyms

symbol	description
T	Test

[symbols, abbreviations, or acronyms — you can simply reference the SRS (**?**) tables, if appropriate —SS]

[Remove this section if it isn't needed —SS]

This document ... [provide an introductory blurb and roadmap of the Verification and Validation plan —SS]

2 General Information

2.1 Summary

[Say what software is being tested. Give its name and a brief overview of its general functions. —SS]

2.2 Objectives

[State what is intended to be accomplished. The objective will be around the qualities that are most important for your project. You might have something like: "build confidence in the software correctness," "demonstrate adequate usability." etc. You won't list all of the qualities, just those that are most important. —SS]

[You should also list the objectives that are out of scope. You don't have the resources to do everything, so what will you be leaving out. For instance, if you are not going to verify the quality of usability, state this. It is also worthwhile to justify why the objectives are left out. —SS]

[The objectives are important because they highlight that you are aware of limitations in your resources for verification and validation. You can't do everything, so what are you going to prioritize? As an example, if your system depends on an external library, you can explicitly state that you will assume that external library has already been verified by its implementation team. —SS]

2.3 Relevant Documentation

[Reference relevant documentation. This will definitely include your SRS and your other project documents (design documents, like MG, MIS, etc). You can include these even before they are written, since by the time the project is done, they will be written. —SS]

?

[Don't just list the other documents. You should explain why they are relevant and how they relate to your VnV efforts. —SS]

3 Plan

[Introduce this section. You can provide a roadmap of the sections to come. —SS]

3.1 Verification and Validation Team

[Your teammates. Maybe your supervisor. You should do more than list names. You should say what each person's role is for the project's verification. A table is a good way to summarize this information. —SS]

3.2 SRS Verification Plan

[List any approaches you intend to use for SRS verification. This may include ad hoc feedback from reviewers, like your classmates, or you may plan for something more rigorous/systematic. —SS]

[Maybe create an SRS checklist? —SS]

3.3 Design Verification Plan

[Plans for design verification —SS]

[The review will include reviews by your classmates —SS]

[Create a checklists? —SS]

3.4 Verification and Validation Plan Verification Plan

[The verification and validation plan is an artifact that should also be verified. Techniques for this include review and mutation testing. —SS]

[The review will include reviews by your classmates —SS]

[Create a checklists? —SS]

3.5 Implementation Verification Plan

[You should at least point to the tests listed in this document and the unit testing plan. —SS]

[In this section you would also give any details of any plans for static verification of the implementation. Potential techniques include code walkthroughs, code inspection, static analyzers, etc. —SS]

3.6 Automated Testing and Verification Tools

[What tools are you using for automated testing. Likely a unit testing framework and maybe a profiling tool, like ValGrind. Other possible tools include a static analyzer, make, continuous integration tools, test coverage tools, etc. Explain your plans for summarizing code coverage metrics. Linters are another important class of tools. For the programming language you select, you should look at the available linters. There may also be tools that verify that coding standards have been respected, like flake9 for Python. —SS]

[If you have already done this in the development plan, you can point to that document. —SS]

[The details of this section will likely evolve as you get closer to the implementation. —SS]

3.7 Software Validation Plan

[If there is any external data that can be used for validation, you should point to it here. If there are no plans for validation, you should state that here. —SS]

[You might want to use review sessions with the stakeholder to check that the requirements document captures the right requirements. Maybe task based inspection? —SS]

[For those capstone teams with an external supervisor, the Rev 0 demo should be used as an opportunity to validate the requirements. You should plan on demonstrating your project to your supervisor shortly after the scheduled Rev 0 demo. The feedback from your supervisor will be very useful for improving your project. —SS]

[For teams without an external supervisor, user testing can serve the same purpose as a Rev 0 demo for the supervisor. —SS]

[This section might reference back to the SRS verification section. —SS]

4 System Test Description

4.1 Tests for Functional Requirements

The following section includes system tests for functional requirements defined in the SRS document. In order to cover all the functional requirements, these subsections match exactly the subsections in the SRS document, which

cover all the different components of this application. These tests will ensure all requirements are fulfilled and the application works as we expected. Most of the following tests will be run manually while some of them will be automated.

4.1.1 Pre-Game Settings

This section includes all test cases related to what users can do before they start to use the application. This will include the requirement for the consent form and user privacy protection (FR-1-1). In order to plan the game, the user must agree to the terms and conditions in the consent form.

1. **FRT-PG1**

   Name: Agree To Consent Form

   Control: Manual

   Initial State: The user is not logged in to the application, and all features in the application are not accessible to the user. A consent form appears asking for access to the device and permission to collect user data

   Input: The user agrees to all the terms and conditions and clicks 'Agree'

   Output: The user is redirected to the login screen

   Test Case Derivation: It is a must to request users' consent before collecting their information or using any hardware component

   How test will be performed: The tester will first clear the app cache and data. Then the tester will run the application, accept the consent form and verify login screen shows up

   Related Requirement(s): FR-1-1

2. **FRT-PG2**

   Name: Disagree To Consent Form

   Control: Manual

   Initial State: The user is not logged in to the application, and all features in the application are not accessible to the user. A consent

form appears asking for access to the device and permission to collect user data

Input: The user rejects the terms and conditions and clicks 'Disagree'

Output: The user is not redirected to the login screen

Test Case Derivation: It is a must to request users' consent before collecting their information or using any hardware component

How test will be performed: The tester will first clear the app cache and data. Then the tester will run the application, reject the consent form and verify login screen does not show up

Related Requirement(s): FR-1-1

4.1.2 User Account

This section includes all test cases related to user accounts. This will include the requirements for account creation (FR-2-1), login (FR-2-3) and deletion (FR-2-2), along with user avatar settings (FR-2-6, FR-2-7). User accounts and virtual avatars are necessary for all users who want to use this application.

1. **FRT-UA1**

   Name: Successful Account Creation

   Control: Manual

   Initial State: The user does not have an account and is not logged in to the application

   Input: All information needed to create an account

   Output: An Account with corresponding information is created in the database with the account initialized to INITIAL_USER_STATE

   Test Case Derivation: User account operations are the foundation of the application. All functionalities work only with an existing account. Also, it is essential to ensure the account information reflects the input information when creating accounts

   How test will be performed: The tester will create an account with all information and verify that the account can be logged into

   Related Requirement(s): FR-2-1

2. **FRT-UA2**

   Name: Unsuccessful Account Creation

   Control: Manual

   Initial State: The user does not have an account and is not logged in to the application

   Input: All information needed to create an account, including an existing username

   Output: Account creation fails with an error message telling the user the username already exists

   Test Case Derivation: Username is the identifier of a user account, which should be unique

   How test will be performed: The tester will create an account with an existing username and verify that the creation fails

   Related Requirement(s): FR-2-1

3. **FRT-UA3**

   Name: Successful Account Login

   Control: Automated

   Initial State: The user has an account and is not logged in to the application

   Input: Username and valid password

   Output: User successfully logs into the application

   Test Case Derivation: User account operations are the foundation of the application. All functionalities work only with an existing account. Also, it is essential to verify that users can log in with only the correct password

   How test will be performed: The tester will create an automated test that inputs a valid password and verifies that the login is completed

   Related Requirement(s): FR-2-3

4. **FRT-UA4**

   Name: Unsuccessful Account Login

Control: Automated

Initial State: The user has an account and is not logged in to the application

Input: Username and wrong password

Output: Login fails with an error message telling the user the password is wrong

Test Case Derivation: User account operations are the foundation of the application. All functionalities work only with an existing account. Also, it is essential to verify that users can log in with only the correct password

How test will be performed: The tester will create an automated test that inputs an invalid password and verifies that a corresponding error is returned

Related Requirement(s): FR-2-3

5. **FRT-UA5**

   Name: Account Deletion

   Control: Manual

   Initial State: The user has an account and is logged into the application

   Input: Corresponding user request

   Output: Data pertaining to the given username is deleted

   Test Case Derivation: User account operations are the foundation of the application. When users want to quit, they should be able to delete all related information in the application by deleting their account

   How test will be performed: The tester will delete the test account and verify the account does not exist anymore in-game

   Related Requirement(s): FR-2-2

6. **FRT-UA6**

   Name: Reset Account Password

   Control: Automated

   Initial State: The user has an account

Input: New password and answers to security questions for password recovery

Output: Password is successfully reset

Test Case Derivation: In case the user misplaces or forgets the password, they can still get their account back

How test will be performed: The tester will first reset the password. Then the tester will try to log in with the new password and verify the new password works properly

Related Requirement(s): FR-2-4

7. **FRT-UA7**

   Name: Human Verification Test

   Control: Manual

   Initial State: The user does not have an account and is not logged into the application

   Input: All information needed to create an account

   Output: The tester passes the test and creates an account successfully

   Test Case Derivation: To prevent malicious automated systems from creating bot accounts

   How test will be performed: The tester will verify the CAPTCHA test appears and functions well when creating an account

   Related Requirement(s): FR-2-5

8. **FRT-UA8**

   Name: Avatar Creation and Modification

   Control: Manual

   Initial State: The user has an account with INITIAL_AVATAR

   Input: Avatar and corresponding user request

   Output: User creates an avatar and changes it

   Test Case Derivation: To help users enjoy social networking when using this application, users shall be able to personalize their account,

therefore it is necessary to verify they can create an avatar and change it whenever they want

How test will be performed: The tester will create an avatar from the initial one and modify it. Then the tester will verify the changes are visible for all friends from another test account

Related Requirement(s): FR-2-6, FR-2-7

4.1.3 Social Networking System

This section includes all test cases related to the social networking system. This will include the requirements for adding (FR-3-1), deleting (FR-3-2), messaging friends of the user (FR-3-3, FR-3-4) and sharing location (FR-3-5). Expanding student social network is the most significant motivation of this application and it is necessary to verify all associated requirements are fully fulfilled.

1. **FRT-SN1**

   Name: Successful Friend Request

   Control: Manual

   Initial State: The user is logged in

   Input: A valid username and corresponding user request

   Output: A Request is sent to the given user

   Test Case Derivation: Users should be able to make friends on this social media platform by searching for names and sending requests

   How test will be performed: The tester will first send a friend request to a test account. Then the tester will verify that test account receives a friend request

   Related Requirement(s): FR-3-1

2. **FRT-SN2**

   Name: Friend Request Acceptance

   Control: Manual

   Initial State: A friend request was sent

   Input: User accepts the request

Output: Two users are added to each other's friend lists

Test Case Derivation: Users should be able to expand networking by accepting friend requests

How test will be performed: The tester will first accept the request. Then the tester will verify that the two accounts become friends of each other

Related Requirement(s): FR-3-1

3. **FRT-SN3**

   Name: Friend Request Rejection

   Control: Manual

   Initial State: A friend request was sent

   Input: User rejects the request

   Output: The request is declined and no friend is added to the list

   Test Case Derivation: Users should be able to decline friend requests from strangers

   How test will be performed: The tester will first reject the request. Then the tester will verify that no friendship relation is generated between the two accounts

   Related Requirement(s): FR-3-1

4. **FRT-SN4**

   Name: Friend Deletion

   Control: Manual

   Initial State: A friend exists in the friend list

   Input: User deletes the chosen friend

   Output: The corresponding friend is deleted from the list

   Test Case Derivation: Users should be able to remove friends from their list to make space for new friends

   How test will be performed: The tester will delete a test friend account and verify the friend is removed from the list

   Related Requirement(s): FR-3-2

5. **FRT-SN5**

   Name: Friend Messaging

   Control: Manual

   Initial State: A friend exists in the friend list

   Input: Audio/text message to send

   Output: The corresponding message is sent to the friend

   Test Case Derivation: Chatting with friends is the core functionality of expanding social networks. Friends should be able to send and receive messages from each other

   How test will be performed: The tester will send an audio and text message to another test friend account. Then the tester will verify the other account received the correct message

   Related Requirement(s): FR-3-3, FR-3-4

6. **FRT-SN6**

   Name: Friend Location Sharing

   Control: Manual

   Initial State: A friend exists in the friend list

   Input: User Location

   Output: User location is visible to the corresponding friend

   Test Case Derivation: Users shall be able to share their current location with friends to meet in person. It is essential to verify the location is accurate and up-to-date

   How test will be performed: The tester will share the location with a friend, and then verify the location is visible and accurate (accuracy of 10 meters) from the other test account

   Related Requirement(s): FR-3-5

4.1.4 AR Campus

This section includes all test cases related to campus navigation with AR technology. This will include the requirements for building recognition (FR-4-1) and demonstration building information (FR-4-2, FR-4-3). Augmented

reality provides an immersive user experience and it is the unique selling point of the application.

1. **FRT-AR1**

   Name: Successful Building Recognition

   Control: Manual

   Initial State: User looks at a building on campus

   Input: Clear Camera view

   Output: The building is recognized and its name is given

   Test Case Derivation: Building recognition is the essential functionality for AR technology applied in this application. The building should be recognizable from a certain angle when users look at their camera

   How test will be performed: The tester will walk around a building on campus and look into the camera, verifying the building is recognized and a list of events/lectures is displayed on the screen

   Related Requirement(s): FR-4-1, FR-4-2, FR-4-3

2. **FRT-AR2**

   Name: Unsuccessful Building Recognition

   Control: Manual

   Initial State: User looks at a building off-campus

   Input: Camera view

   Output: The building is not recognized correctly

   Test Case Derivation: Building recognition is the essential functionality for AR technology applied in this application. The building should be recognizable from a certain angle when users look at their camera

   How test will be performed: The tester will walk around a building that is not on campus and look into the camera, verifying the building is not recognized or the system recognizing it as a school building

   Related Requirement(s): FR-4-1, FR-4-2, FR-4-3

4.1.5 Lectures and Events

This section includes all test cases related to events and lectures happening on campus. This will include the requirements for events and lectures themselves, users interacting with these properties, and the power of administration accounts (All FR-5 requirements in the SRS document). The test cases ensure that the lectures and events information is accurate and helpful for all users.

1. **FRT-LE1**

   Name: Interest/Disinterest Event

   Control: Manual

   Initial State: An event exists in a specific building

   Input: Corresponding user request

   Output: The event with all necessary information is added/removed from the user's event list

   Test Case Derivation: The user should be able to interact with events available on campus and share their activities with friends. Therefore, they should be able to switch between 'interest' and 'disinterest' states for all events

   How test will be performed: The tester will pin and unpin the event and verify the event shows up and disappears from the user's interested event list. Then the tester will verify the event has all associated information including club/department, location and time

   Related Requirement(s): FR-5-1, FR-5-2, FR-5-7

2. **FRT-LE2**

   Name: Pin/unpin Lectures

   Control: Manual

   Initial State: A lecture with all necessary information exists in a specific building

   Input: Corresponding user request

   Output: The lecture is added/removed from the user's lecture list

Test Case Derivation: The user should be able to interact with lectures shown in the app and share their schedule with friends. Therefore, they should be able to switch between 'pinned' and 'unpinned' states for all lectures

How test will be performed: The tester will pin and unpin the lecture and verify the lecture shows up and disappears from the user's pinned lecture list. Then the tester will verify the event has all associated information including instructor, location and time

Related Requirement(s): FR-5-3, FR-5-4, FR-5-8

3. **FRT-LE3**

   Name: Administrator Add Events

   Control: Manual

   Initial State: User is logged in as an admin

   Input: Sample event with all necessary information

   Output: The event is posted on the building and is available for all users to pin

   Test Case Derivation: Administrators work as a source of truth in this application, therefore they shall be able to create new events for all users

   How test will be performed: The tester will log in as an administrator and add the sample event. Then the tester will verify the event shown in the corresponding building is available for all users

   Related Requirement(s): FR-5-5

4. **FRT-LE4**

   Name: Administrator Change Events

   Control: Manual

   Initial State: User is logged in as an admin

   Input: New event information and corresponding user request

   Output: The posted event information is updated

   Test Case Derivation: Administrators work as a source of truth in this application, therefore they shall be able to update event information to get users noticed

How test will be performed: The tester will log in as an administrator and update the sample event. Then the tester will verify the event shown in the corresponding building is updated for all users

Related Requirement(s): FR-5-5

5. **FRT-LE5**

   Name: Administrator Delete Events

   Control: Manual

   Initial State: User is logged in as an admin

   Input: Existing event and corresponding user request

   Output: The event is deleted

   Test Case Derivation: Administrators work as a source of truth in this application, therefore they shall be able to clean out-of-date event

   How test will be performed: The tester will log in as an administrator and delete the sample event. Then the tester will verify the event shown in the corresponding building is deleted for all users

   Related Requirement(s): FR-5-5

6. **FRT-LE6**

   Name: Administrator Add Lectures

   Control: Manual

   Initial State: User is logged in as an admin

   Input: Sample lecture with all necessary information

   Output: The lecture is posted on the building and is available for all users to pin

   Test Case Derivation: Administrators work as a source of truth in this application, therefore they shall be able to create new lectures for all users when new semesters come

   How test will be performed: The tester will log in as an administrator and add the sample lecture. Then the tester will verify the lecture shown in the corresponding building is available for all users

   Related Requirement(s): FR-5-6

7. **FRT-LE7**

   Name: Administrator Change Lecture

   Control: Manual

   Initial State: User is logged in as an admin

   Input: New lecture information and corresponding user request

   Output: The posted lecture information is updated

   Test Case Derivation: Administrators work as a source of truth in this application, therefore they shall be able to update lecture information to get users noticed

   How test will be performed: The tester will log in as an administrator and update the sample lecture. Then the tester will verify the lecture shown in the corresponding building is updated for all users

   Related Requirement(s): FR-5-6

8. **FRT-LE8**

   Name: Administrator Delete Lectures

   Control: Manual

   Initial State: User is logged in as an admin

   Input: Existing lecture and corresponding user request

   Output: The lecture is deleted

   Test Case Derivation: Administrators work as a source of truth in this application, therefore they shall be able to clean out-of-date lectures after each semester

   How test will be performed: The tester will log in as an administrator and delete the sample lecture. Then the tester will verify the lecture shown in the corresponding building is deleted for all users

   Related Requirement(s): FR-5-6

4.2 Tests for Nonfunctional Requirements

The following section includes tests for non-functional requirements defined in the SRS document. Areas of testing follow the subsections of requirements in the previous document, which mainly include Look and Feel, Usability and

Humanity, Performance and Security.
Usability requirements will be tested by asking users to do a survey, whose content can be found in section 6.2. Most of the tests here are dynamic and will be done manually or automatically, but there are some tests that need non-dynamic testing like peer code review or code walkthrough.

4.2.1 Look and Feel

1. **FRT-LF1**

   Name: Survey for feedback on application layout

   Type: Non-functional, Dynamic, Manual

   Initial State: User has an account

   Input/Condition: User is logged into the homepage and pokes around the application

   Output/Result: Get feedback and verify layout is user-friendly. This test is a pass if average individual score is over MIN_SCORE in the "User-friendly Layout", "Intuitive icons" and "Immediate Visual Response when Clicking" categories of the User Experience Survey

   How test will be performed: A survey will be given to at least SURVEY_SAMPLE_SIZE users where they will give feedback to different elements of the interface. All of the survey takers are selected randomly from McMaster University

   Related Requirement(s): LF-A1, LF-A2

2. test-id1

   Type: Functional, Dynamic, Manual, Static etc.

   Initial State:

   Input/Condition:

   Output/Result:

   How test will be performed:

3. test-id2

Type: Functional, Dynamic, Manual, Static etc.

Initial State:

Input:

Output:

How test will be performed:

4.2.2 Area of Testing2

...

4.3 Traceability Between Test Cases and Requirements

[Provide a table that shows which test cases are supporting which requirements. —SS]

5 Unit Test Description

[This section should not be filled in until after the MIS (detailed design document) has been completed. —SS]

[Reference your MIS (detailed design document) and explain your overall philosophy for test case selection. —SS]

[To save space and time, it may be an option to provide less detail in this section. For the unit tests you can potentially layout your testing strategy here. That is, you can explain how tests will be selected for each module. For instance, your test building approach could be test cases for each access program, including one test for normal behaviour and as many tests as needed for edge cases. Rather than create the details of the input and output here, you could point to the unit testing code. For this to work, you code needs to be well-documented, with meaningful names for all of the tests. —SS]

5.1 Unit Testing Scope

[What modules are outside of the scope. If there are modules that are developed by someone else, then you would say here if you aren't planning on verifying them. There may also be modules that are part of your software, but have a lower priority for verification than others. If this is the case, explain your rationale for the ranking of module importance. —SS]

5.2 Tests for Functional Requirements

[Most of the verification will be through automated unit testing. If appropriate specific modules can be verified by a non-testing based technique. That can also be documented in this section. —SS]

5.2.1 Module 1

[Include a blurb here to explain why the subsections below cover the module. References to the MIS would be good. You will want tests from a black box perspective and from a white box perspective. Explain to the reader how the tests were selected. —SS]

1. test-id1

   Type: [Functional, Dynamic, Manual, Automatic, Static etc. Most will be automatic —SS]

   Initial State:

   Input:

   Output: [The expected result for the given inputs —SS]

   Test Case Derivation: [Justify the expected value given in the Output field —SS]

   How test will be performed:

2. test-id2

   Type: [Functional, Dynamic, Manual, Automatic, Static etc. Most will be automatic —SS]

   Initial State:

   Input:

   Output: [The expected result for the given inputs —SS]

   Test Case Derivation: [Justify the expected value given in the Output field —SS]

   How test will be performed:

3. ...

5.2.2 Module 2

...

5.3 Tests for Nonfunctional Requirements

[If there is a module that needs to be independently assessed for performance, those test cases can go here. In some projects, planning for nonfunctional tests of units will not be that relevant. —SS]

[These tests may involve collecting performance data from previously mentioned functional tests. —SS]

5.3.1 Module ?

1. test-id1

   Type: [Functional, Dynamic, Manual, Automatic, Static etc. Most will be automatic —SS]

   Initial State:

   Input/Condition:

   Output/Result:

   How test will be performed:

2. test-id2

   Type: Functional, Dynamic, Manual, Static etc.

   Initial State:

   Input:

   Output:

   How test will be performed:

5.3.2 Module ?

...

5.4 Traceability Between Test Cases and Modules

[Provide evidence that all of the modules have been considered. —SS]

6 Appendix

This is where you can place additional information.

6.1 Symbolic Parameters

The definition of the test cases will call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance.

Table 1: **Symbolic Parameter Table**

Symbolic Parameter	**Description**	**Value**
INITIAL_USER_STATE	The default user state with all default user information	All entries empty except username and password
INITIAL_AVATAR	The default virtual avatar	An unisex virtual avatar with default settings
MIN_SCORE	The passing grade for a category in the survey	7/10
SURVEY_SAMPLE_SIZE	Size of the user experience survey	50

6.2 Usability Survey Questions?

[This is a section that would be appropriate for some projects. —SS]

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning. Please answer the following questions:

Appendix — Reflection

[This section is not required for CAS 741 —SS]

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning. Please answer the following questions:

1. What knowledge and skills will the team collectively need to acquire to successfully complete the verification and validation of your project? Examples of possible knowledge and skills include dynamic testing knowledge, static testing knowledge, specific tool usage etc. You should look to identify at least one item for each team member.

2. For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?