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

November 23, 2023

Revision History

Date	Version	Notes
Oct 29	1.0	Add Functional Requirements Tests
Oct 31	1.0	Add Non-Functional Requirement Tests
Nov 1	1.0	Add Unit Tests
Nov 3	1.0	Add Tables And More Tests
Nov 22	1.0	Revision 0: Resolve issues

Contents

List of Tables

1 Symbols, Abbreviations, and Acronyms

symbol	**description**
VnV	Verification and Validation
HA	Hazard Analysis
UI	User Interface
AR	Augmented Reality
SRS	Software Requirement Specification
CAPTCHA	Human Verification Test
MIS	Module Interface Specification
Vuforia	AR software development kit
FEMA	Failure Mode and Effect Analysis
UTF	Unity Test Framework
APK	Android Package, a file format used by Android
GPS	Global Positioning System
Apache JMeter	A load testing tool for performance analysis

This document describes the verification and validation plan for CampusConnection. Within this document, there will be plans to verify the SRS document, the design plan, the VnV plan, the implementation plan, and the software validation plan. Furthermore, this document will list specific system tests that cover all functional and non-function requirements we mentioned in the SRS, along with a traceability table between test cases and requirements to bind all requirements with tests.

This document also includes unit tests, but since the design documentation like MIS is not ready yet, we will only cover the most important modules and its corresponding methods test. The team will iterate on the VnV plan and have a more detailed version with test code when the modules are specified in the design document.

2 General Information

2.1 Summary

The software being tested is a social media application which also uses Augmented Reality features to allow McMaster University students to connect with each other. The software has many different functions to be able to accomplish its goals. These are some of the important functions:

- Allowing the users to create and manage their account.
- Allowing the users to add, manage and message friends.
- Allowing the users with administrator access to add and manage events and lectures.
- Allowing the users to view ongoing events and pin and unpin them based on interest.
- Identifying major campus buildings through the user's location and camera view, and using the information to display relevant lectures, events and their timings.

2.2 Objectives

2.2.1 Important Qualities

In order to develop the software to meet its goals and create a complete product, there are many qualities that are important for our project such as:

- An easily navigable User Interface adaptable to different screen sizes
- Sufficient speed and reliability of the building recognition features
- Sufficient speed and reliability in sending and receiving messages
- Adequate accuracy in tracking the user's location
- Appropriate abidance of security requirements regarding critical information such as location and personal data.
- Infrastructure to support a passable scale of total concurrent users and total user data storage

2.2.2 Out of Scope Objectives

While we would want to satisfy as many positive qualities as possible, due to limited resources , there will be some objectives that we will be leaving out since they are out of scope such as :

- Project longevity requirements:

  This is because the longevity requirements were written with the assumption that the product will be supported even after the duration of the course for years to come which is something that is not possible to verify or guarantee and is out of the scope. Instead, we could have a code inspection with other developers and the supervisor about these requirements to gain some confidence.

- Testing learning requirements:

  We will be producing material to make it easier to understand and use our product but to test every piece of supporting documentation and tutorial content would require long testing sessions with volunteers which can be out of the scope. Instead, we could use the feedback obtained from the survey to improve our learning specific content.

2.2.3 Priority of Objectives

We will be prioritizing reliability in our listed qualities over performance since we want our software to be correct and consistent over fast and unreliable. While we still want to maximize performance where possible, it is difficult to predict exactly how well we will meet our performance criteria since we will be limited by many factors out of our control such as external libraries.

We will also be assuming that any external libraries we depend on (such as Vuforia) will have already been verified by the implementation team due to our lack of control over them.

2.3 Relevant Documentation

Many of the other documents associated with our project are relevant to the Verification and Validation plan such as:

- Development Plan:

  The Development Plan contains the roles and responsibilities of each member, which outlines the tester of each components. It also includes the tools and technologies that will be used to facilitate testing

  Development Plan

- Software Requirement Specification (SRS):

  The SRS has a lot of components relevant to the VnV plan. The functional requirements are important in identifying the primary functions of the the project while the non functional requirements are critical in identifying the main qualities and objectives of the product. It also contains constraint and assumption information which can help identify out-of-scope objectives.

  Software Requirement Specification (SRS)

- Hazard Analysis (HA):

  The HA document contains the Failure Mode and Effect Analysis (FMEA) which is incredibly useful in understanding the possible failures and their consequences. This can help us assign priorities to different failures that can be translated into priorities for different qualities identified earlier in the document. It also contains more detailed safety and security requirements that we can use to verify our project.

Hazard Analysis

3 Plan

3.1 Verification and Validation Team

Team Member	Role in Verification and Validation
Abhiram Neelamraju	**Lead tester**, Leads the testing process and generate test reports
Michael Kim	**User Experience Tester**, Tests the user interface of the software to assess how intuitive the AR aspects are.
Firas Elayan	**Validation analyst**, Analyses test results against standards and project goals to ensure alignment.
Zihao Du	**Performance tester**, Runs tests and analyses to evaluate the performance of the software system.
Matthew Miller	**Feedback coordinator**, Manages the feedback gathered through reviews conducted by different parties as well as during testing to ensure the feedback is taken into account during development.
Waseef Nayeem	**Issue resolver**, Deal with bugs and issues that arise during the development and validation of the software.

3.2 SRS Verification Plan

We will be following a few approaches to verify our SRS document, establishing a strong base for the development of our project.

- Ad-hoc review by classmates:

  We will be conducting unstructured reviews of our SRS document with our classmates. A couple of different teams of students will go through

the document to give us feedback and improvement suggestions around its contents and formatting by creating GitHub issues. This has been done before for Revision 0 and we decide to continue verifying the document for Revision 1.

- Ad-hoc review by our supervisor:

  Our supervisor, Dr. Yuan will perform a review of our SRS document by reading through it and giving us feedback on it. Dr. Yuan's expertise would provide us with valuable insight, allowing us to improve the quality and correctness of the SRS. Given her experience as well, it would ensure we're properly following set standards.

- SRS walkthrough by team:

  A walkthrough of the document will be performed together by the members of our team. We will all go through each section and analyse it, where we'l lbe able to suggest changes and improvements. Different team members worked on different parts of the document, so this walkthrough will allow for new perspectives on each section.

3.3 Design Verification Plan

- Creating mock-ups and prototypes:

  In order to carry out good and efficient evaluations of our design, we will create design mock-ups and prototypes. These will be lo-fi at first and increase in fidelity as the software development progresses. The prototypes will help improve the reviews and feedback of the design by providing a visual (possibly physical) representation of our design, helping reviewers judge the features and quality of the design.

- Requirements checklist:

  A checklist of our requirements from the SRS document will be made to be used as part of the verification of our design. The checklist will be used during team reviews in order to ensure our design meets the project requirements.

- Supervisor review:

As with the SRS document, our designs will receive an ad hoc review by our supervisor, Dr. Yuan. She will ensure our designs meet software standards and adhere to relevant principles.

- Classmates review:

  We will conduct reviews with our classmates to get their input on our designs. This is very valuable for us as our classmates fall under one of our stakeholders, the customers.

3.4 Verification and Validation Plan Verification Plan

- Team review:

  Our team members will review the VnV plan together to ensure the plan is complete and correct in how/what it verifies and validates. The review will allow us to make sure the VnV plan is line with the goals set for our project.

- Supervisor review:

  As with the SRS document and our designs, our supervisor, Dr. Yuan, will conduct a review of our VnV plan. Her experience and knowledge of software engineering will allow her to find any shortcomings in our plans and to provide very valuable feedback.

- Classmates review:

  Our classmates will conduct their own review of our VnV plan. As they resemble our stakeholders and each have their own experience and skills, their feedback can prove to be very beneficial and can provide a range of perspectives on the effectiveness our plan.

- Mutation testing:

  In order to determine how effective our outlined tests in the VnV plan are at ensuring the correctness of our product, we will employ mutation testings. Mutations will be introduced to the parts of our source code that is covered by the test cases outlined here. These mutations would be very small but should cause the tests to fail. This would allow us to determine how effective our tests are at finding faults and flaws in our product's code.

3.5 Implementation Verification Plan

Our implementation verification plan involves making use of the system and unit tests we've outlined in the VnV plan to ensure the correctness and quality of our implementation of our product.

We've outlined system tests and unit tests for our requirements. The system tests apply to the application as a whole, placing a focus on verifying the correctness of the application and how well it satisfies the requirements we've outlined in our SRS document. Our unit tests focus on specific modules within our application, aiming at assessing the accuracy of each module, both individually and when working together. All our tests are based on our functional and non-functional requirements to ensure our application adheres to our goals for the project.

In addition to the tests outlined in this document, we will also employ some static technqiues to verify our implementation. We will be making use of verification tools that are able to identify errors and issues in our source code. We will also be conducting code reviews, where we'll be examining our code (without executing it) to find any problems in relation with our code's adherence to our design and requirements.

3.6 Automated Testing and Verification Tools

We intend to make use of several different tools that will help us in the testing and verification of our product. We have outlined some of these tools in our development plan.

Since we are coding our product using C#, we will be making use of JetBrains Rider's in-built linter for C#. Linters are incredibly useful tools that we can use to detect any programming errors and bugs in our source code, as well as any issues with the style. Using this linter would help avoid these problems growing as we build the project, ensuring its correctness.

As we are using Unity as our engine to develop our application, our testing framework will be the Unity Test Framework. Unity Test Framework can be used for both automated testing (Play Mode) and unit testing tool (edit mode) and there are other UI driven end to end automated tool that

we can use like AltTester if it is too complicated to write automated testing code in Unity Test Framework.

3.7 Software Validation Plan

We will be conductions formal reviews with some of our stakeholders as well as our supervisor, Dr. Yuan. These reviews will ensure that our application meets the requirements as well as the goals of our stakeholders. Dr. Yuan will make sure the product we've build meets different standards.

Moreover, we'll use task-based inspection, involving testing different aspects of our app through real-world scenarios. This will aid us in evaluating the functionality of our software. In addition, we will also have students - who represent our stakeholders - test out features of our software, which will allow us to observe our software's performance in the real world. The feedback we get from user testing would be very helpful in ensuring the alignment of our software with the user goals and requirements.

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), deletion (FR-2-2) and email verification(FR-2-8), 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 (username and password)

   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 with 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: User clicks on the delete account button

   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 (username and password)

   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 user clicks on modifying avatar button

   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

9. **FRT-UA9**

   Name: Email verification

   Control: Manual

Initial State: The user adds McMaster email to user profile

Input: User verifying after receving the verification email

Output: User email is verified and all access is granted for the account

Test Case Derivation: The user has to be identified as a McMaster student to have enough access to all functionalities (Event and Lecture information). So it is essential to test the email verification functionality

How test will be performed: The tester will add a test McMaster email to the test account and verify the email can be verified and all functionalities work for the account after email verification

Related Requirement(s): FR-2-8

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 GPS_ACCURACY) 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: User clicks on the corresponding button of the event

   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: User clicks on the corresponding button of the lecture

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 name, department, time and location

   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 the admin clicks on modify event button

Output: The posted event information is changed to the new one

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 the admin clicks on delete event button

   Output: The event is deleted and no user can see it from their end any more

   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 name, instructor, time and location

   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 the admin clicks on modify lecture button

   Output: The posted lecture information is changed to the new one

   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 and the admin clicks on delete lecture button

   Output: The lecture is deleted and no user can see it from their end any more

   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.
This section also includes new requirements for the HA, but when they are added to the SRS, they are given new IDs for consistency, please check the HA for more details.
Usability requirements will be tested by asking users to do a survey, whose content can be found in section 7.2. 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. **NFRT-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. The average individual score is over MIN_SCORE in the "User-friendly Layout", "Intuitive icons", "Immediate Visual Response when Clicking" and "Appealing Colour Scheme" 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, LF-S2

2. **NFRT-LF2**

   Name: Visual Inspection with different screen sizes

   Type: Non-functional, Dynamic, Manual

   Initial State: User has the application downloaded on their phone

   Input/Condition: User has the application open on their phone

   Output/Result: For all different pages all visual elements on the screen are within the borders of the screen for all screens in the SCREEN_VIEWPORTS list

   How test will be performed: The application will be opened on all screens in the SCREEN_VIEWPORTS list. The testers will visually inspect each page and each screen to make sure that all visual elements on the screen are within its borders

   Related Requirement(s): LF-A3

3. **NFRT-LF3**

   Name: Visual inspection of color scheme

   Type: Non-functional, Dynamic, Manual

   Initial State: User has the application downloaded on their phone

   Input/Condition: User has the application open on their phone

   Output/Result: For all different pages the colour scheme is the same

   How test will be performed: The testers will visually inspect each page to make sure that the colour scheme on each page is the same

   Related Requirement(s): LF-S1

4.2.2 Usability and Humanity

1. **NFRT-UH1**

   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. The average individual score is over MIN_SCORE in the "Easy to Navigate", "Helpful Tutorial" and "No Technical or Software-Specific Language" 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): UH-EOU1, UH-L2, UH-UP1

2. **NFRT-UH2**

   Name: Visual inspection of user profile

   Type: Non-functional, Dynamic, Manual

   Initial State: User has the application downloaded on their phone

   Input/Condition: User has the application open on their phone

   Output/Result:

   - The name, time, and location of all pinned events and lectures are displayed on a user's personal page
   - The user is able to update their USER_PROFILE and avatar

   How test will be performed: The testers will open the application on one device and go to the user profile page. The testers will visually inspect the page to check that the above conditions are satisfied.

   Related Requirement(s): UH-PI1, UH-PI2

3. **NFRT-UH3**

   Name: Visual inspection of tutorial

   Type: Non-functional, Dynamic, Manual

   Initial State: User has the application downloaded on their phone

   Input/Condition: User has the application open on their phone

Output/Result: A tutorial explaining the features of the application appears on first launch, and is available upon request from the user

How test will be performed: The testers will open the application on one device and visually inspect the application to check that the tutorial appears on first launch and is available upon request

Related Requirement(s): UH-L1

4.2.3 Performance

1. **NFRT-P1**

   Name: Weak network strength Notification

   Type: Non-functional, Dynamic, Manual

   Initial State: User has the application downloaded on their phone

   Input/Condition: User enters a zone with week internet connection

   Output/Result: A notification in the application appears saying the network strength weak or the server connection is poor

   How test will be performed: The testers will open the application on one device and visually inspect the application sending a message when the network strength is weak

   Related Requirement(s): P-RF1

2. **NFRT-P2**

   Name: Error Message when internet connection is lost

   Type: Non-functional, Dynamic, Manual

   Initial State: User has the application downloaded on their phone

   Input/Condition: User loses internet connection

   Output/Result: An error message stating that there is no internet connection is displayed

   How test will be performed: The testers will open the application on one device and visually inspect the application to check the error message appears when the application fails to connect to the internet

   Related Requirement(s): P-RF3

3. **NFRT-P3**

   Name: AR incompatibility Warning

   Type: Non-functional, Dynamic, Manual

   Initial State: User has the application downloaded on their phone which is not compatible with AR features

   Input/Condition: User uses the AR feature

   Output/Result: A warning message is displayed telling the user AR is not available

   How test will be performed: The testers will open the application on a device not compatible with AR functions and visually inspect the error message appears when AR feature is used

   Related Requirement(s): P-RF5

4. **NFRT-P4**

   Name: Message when starting AR

   Type: Non-functional, Dynamic, Manual

   Initial State: User has the application downloaded on their phone

   Input/Condition: User uses the AR feature

   Output/Result: A message telling the user to be aware of their surroundings is displayed upon startup.

   How test will be performed: The testers will open the AR feature on one device and visually inspect the application to check that the message is shown

   Related Requirement(s): P-RF4

5. **NFRT-P5**

   Name: Rudimentary functions when internet connection is lost

   Type: Non-functional, Dynamic, Manual

   Initial State: User has no internet connection

   Input/Condition: User opens the application

   Output/Result: The application still works with limited functionalities (personal profile and events view)

How test will be performed: The testers will open the application with offline mode and check the personal profile and pinned events/lectures are still available

Related Requirement(s): P-RF6

6. **NFRT-P6**

   Name: Builing Recognition Response Time

   Type: Non-functional, Dynamic, Manual

   Initial State: The product is installed

   Input/Condition: The user opens the AR feature and points the camera to a building on campus

   Output/Result: The popup response following successful image recognitions appears within RECOGNITION_TIME

   How test will be performed: The tester will start a timer when the recognition process starts and end the timer when the recognition is successful. Then the tester can verify if the time is within RECOGNITION_TIME

   Related Requirement(s): P-SL1

7. **NFRT-P7**

   Name: Message Sent Response Time

   Type: Non-functional, Dynamic, Manual

   Initial State: Two test accounts are friends of each other

   Input/Condition: A message is sent from one account to the other

   Output/Result: The user receives a message within MESSAGE_TIME after being sent by another user

   How test will be performed: The tester will start a timer when the message is sent and end the timer when the message is received. Then the tester can verify if the time is within MESSAGE_TIME

   Related Requirement(s): P-SL2

8. **NFRT-P8**

   Name: Location Sharing Update Time

Type: Non-functional, Dynamic, Manual

Initial State: Two test accounts are friends of each other, one account starts location sharing

Input/Condition: The device sharing its location moves in a period of time

Output/Result: The user's updated location must appear on their friends' device within LOCATION_UPDATE_TIME of the location change

How test will be performed: The tester will start a timer when the device moves and end the timer when the location change is shown on the other device. Then the tester can verify if the time is within LOCATION_UPDATE_TIME

Related Requirement(s): P-SL3

9. **NFRT-P9**

   Name: Messaging When Location Sharing Is Not Working

   Type: Non-functional, Dynamic, Manual

   Initial State: Two test accounts are friends of each other, one of them cannot share location

   Input/Condition: The one device that cannot share location messages the other one

   Output/Result: The message is sent successfully

   How test will be performed: The tester will turn off the location permission and send a message. Then the tester will verify the message is sent without a problem

   Related Requirement(s): P-RF2

10. **NFRT-P10**

    Name: Code Walkthrough For User Personal Data

    Type: Non-functional, Static, Manual

    Input/Condition: Source code

    Output/Result: Successfully convinced the stakeholders and users the following:

- User's personal information does not appear in the database if the user did not grant permission

How test will be performed: A code walkthough will be held by the developers to show that database storing is only possible when the user grants permission

Related Requirement(s): P-SC1

11. **NFRT-P11**

    Name: Code Inspection For Legitimate Use Of Personal Data

    Type: Non-functional, Static, Manual

    Input/Condition: Source code

    Output/Result: Successfully convinced the stakeholders and users the following:

    - The usage of a user's persona information by the product abides by the Privacy Act, The Personal Information Protection and Electronic Documents Act, and Canada and Ontario's data protection laws

    How test will be performed: A code inspection will be held with supervisor's attendance to show the usage of personal data is legitimate

    Related Requirement(s): P-SC3

12. **NFRT-P12**

    Name: Code Walkthrough For Data Transmission

    Type: Non-functional, Static, Manual

    Input/Condition: Source code

    Output/Result: Successfully convinced the stakeholders and users the following:

    - The product does not execute any code that involves the transmission of information outside of the product

    How test will be performed: A code walkthough will be held by the developers to show that the application does not transmit data while not in use

Related Requirement(s): P-SC4

13. **NFRT-P13**

    Name: Code Walkthrough For Database

    Type: Non-functional, Static, Manual

    Input/Condition: Source code

    Output/Result: Successfully convinced the stakeholders and users the following:

    - Information for an event added to the database is still present at least 3 months after its addition

    How test will be performed: A code walkthough will be held by the developers to show that the database is reliable and the tables will be kept for at least 3 months

    Related Requirement(s): P-C2

14. **NFRT-P14**

    Name: Code Peer Evaluation For Longevity

    Type: Non-functional, Static, Manual

    Input/Condition: Source code

    Output/Result: Successfully convinced the stakeholders and users the following:

    - The product is able to operate without major malfunctions in release build for at least 1 year
    - The finalized product will remain compatible with the promised operating systems and devices for at least 3 years

    How test will be performed: The testers will hold a peer evaluation session for the other developers to inspect the source code and make sure it can operate and remain compatible with promised operating systems for a long time

    Related Requirement(s): P-L1, P-L2

15. **NFRT-P15**

    Name: New Building Addition

    Type: Non-functional, Dynamic, Manual

    initial State: The tester has developer permission

    Input/Condition: All information about the new building (location, name, image, etc.)

    Output/Result: A new building is added without huge drawbacks in performance

    How test will be performed: The tester will add a dummy new building and test if the all the time requirements mentioned in the Speed and Latency Requirements (P-SL1, P-SL2, P-SL3) are still fulfilled.

    Related Requirement(s): P-SE3

16. **NFRT-P16**

    Name: Building Identification

    Type: Non-functional, Dynamic, Manual

    Initial State: The user has the product launched on their device

    Input/Condition: The product is scanning a building on campus

    Output/Result: The product identifies a building with a success rate of at least 80 percent

    How test will be performed: Testers will scan buildings with the product multiple times and calculate the percentage of successful scans

    Related Requirement(s): P-PA1

17. **NFRT-P17**

    Name: Load Testing for Server Capacity

    Type: Non-functional, Dynamic, Automatic

    Initial State: The server is online and ready to connect to

    Input/Condition: Load testing tool simulates there are MAX_CAPACITY users simultaneously accessing the server

Output/Result: The server is able to handle all the users connecting to the server at once

How test will be performed: Testers use Apache JMeter for the load test simulation

Related Requirement(s): P-C1, P-SE1

18. **NFRT-P18**

    Name: Load Testing for Database Capacity

    Type: Non-functional, Dynamic, Automatic

    Initial State: The database is online and ready to connect to

    Input/Condition: Load testing tool simulates there is information for MAX_USER_LOAD users stored in the database

    Output/Result: The database can handle all the data that it contains

    How test will be performed: Testers use Apache JMeter for the load test simulation

    Related Requirement(s): P-SE2

4.2.4 Operational and Environmental

1. **NFRT-OE1**

   Name: Visual inspection for application download

   Type: Non-functional, Manual

   Initial State: User has a phone that uses Android 11 or above

   Input/Condition: User's phone is turned on and unlocked

   Output/Result: The product can be downloaded onto the phone from the Google Play Store, or by downloading the APK file directly

   How test will be performed: The testers will check the Google Play Store for the product, or find an APK file for the product. The testers will then download and install the product onto the phone from the Google Play Store or the APK file.

   Related Requirement(s): OE-P1

4.2.5 Maintainability and Support

1. **NFRT-MS1**

   Name: Survey For Maintenance Time

   Type: Non-functional, Static, Manual

   Initial State: NA

   Input/Condition: Ask at least 50 users when they are less likely to use the application and collect the feedback

   Output/Result: Periods of reduced usage is shown to arrange updates

   How test will be performed: Testers ask at least 50 users when is a reduced usage period for them, and schedule coming updates during these periods

   Related Requirement(s): MS-M1

2. **NFRT-MS2**

   Name: Peer Evaluation for Feature Request

   Type: Non-functional, Static, Manual

   Initial State: A public Git repo for this application

   Input/Condition: An issue is created

   Output/Result: All functionalities of this repo work normally

   How test will be performed: Testers will ask for a peer evaluation on this application repo, and the other developers will verify there is a user manual and issues can be used for feature request or bug report

   Related Requirement(s): MS-S1, MS-S2

3. **NFRT-MS3**

   Name: Android Version Test

   Type: Non-functional, Dynamic, Manual

   Initial State: The application is installed on devices with Android 11 and above version

   Input/Condition: The application is opened

Output/Result: The application works normally

How test will be performed: Testers will install the application on different Android version (11 and above) and verify the application can start without errors, and the core features (Map, friend chatting and AR camera) have the same performance with different Android versions.

Related Requirement(s): OE-EPE1, OE-IAS1, MS-A1

4.2.6 Security

1. **NFRT-S1**

   Name: Access Test

   Type: Non-functional, Dynamic, Manual

   Initial State: The product is downloaded onto a supported device

   Input/Condition: The product is launched

   Output/Result: At each level of access, the product constrains the possible actions to what is specified in requirement S-A1.

   How test will be performed:

   (a) The tester will download the product to a supported device and launch the product.

   (b) The product will begin at the first level of access, which should only allow the user to view the login, account creation, and account recovery pages.

   (c) The tester will log in or create an account to move to the second level of access (McMaster student/faculty member).

   (d) The tester will test that actions only possible at the third level of access (adding/deleting/editing events, pull logs not accessible to users) are not possible to do at this level.

   (e) The tester will log out of the account and log in as an administrator, moving to the third level of access.

   (f) The tester will test that actions only possible at this level of access (adding/deleting/editing events, pull logs not accessible to users) are possible to do.

   Related Requirement(s): S-A1

4.2.7 Cultural

1. **NFRT-CUL1**

   Name: Survey for feedback on cultural requirements

   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 not culturally offensive. This test is a pass if average individual score is over MIN SCORE in the "Cultural Friendliness" 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): CUL-C1, CUL-C2

4.2.8 Compliance

N/A

4.3 Traceability Between Test Cases and Requirements

Table 1: Traceability Between Functional Test Cases and Functional Requirements, FR-1 to FR-3

Test IDs	**Functional Requirement IDs**													
	FR1-1	**FR2-1**	**FR2-2**	**FR2-3**	**FR2-4**	**FR2-5**	**FR2-6**	**FR2-7**	**FR2-8**	**FR3-1**	**FR3-2**	**FR3-3**	**FR3-4**	**FR3-5**
FRT-PG1	X													
FRT-PG2	X													
FRT-UA1		X												
FRT-UA2		X												
FRT-UA3				X										
FRT-UA4				X										
FRT-UA5			X											
FRT-UA6					X									
FRT-UA7						X								
FRT-UA8							X	X						
FRT-UA9									X					
FRT-SN1										X				
FRT-SN2										X				
FRT-SN3										X				
FRT-SN4											X			
FRT-SN5												X	X	
FRT-SN6														X

Table 2: Traceability Between Functional Test Cases and Functional Requirements, FR-4 to FR-5

Test IDs	**Functional Requirement IDs**										
	FR4-1	**FR4-2**	**FR-4-3**	**FR5-1**	**FR5-2**	**FR5-3**	**FR5-4**	**FR5-5**	**FR5-6**	**FR5-7**	**FR5-8**
FRT-AR1	X	X	X								
FRT-AR2	X	X	X								
FRT-LE1				X	X					X	
FRT-LE2						X	X				X
FRT-LE3								X			
FRT-LE4								X			
FRT-LE5								X			
FRT-LE6									X		
FRT-LE7									X		
FRT-LE8									X		

Table 3: Traceability Between Nonfunctional Test Cases and Nonfunctional Requirements, LF-A1 to UH-A1

Test IDs	**Nonfunctional Requirement IDs**											
	LF-A1	**LF-A2**	**LF-A3**	**LF-S1**	**LF-S2**	**UH-EOU1**	**UH-PI1**	**UH-PI2**	**UH-L1**	**UH-L2**	**UH-UP1**	**UH-A1**
NFRT-LF1	X	X			X							
NFRT-LF2			X									
NFRT-LF3				X								
NFRT-UH1						X				X	X	
NFRT-UH2							X	X				
NFRT-UH3	X								X			

Table 4: Traceability Between Nonfunctional Test Cases and Nonfunctional Requirements, P-SL1 to P-RF4

Test IDs	**Nonfunctional Requirement IDs**												
	P-SL1	**P-SL2**	**P-SL3**	**P-SC1**	**P-SC2**	**P-SC3**	**P-SC4**	**P-PA1**	**P-PA2**	**P-RF1**	**P-RF2**	**P-RF3**	**P-RF4**
NFRT-P1										X			
NFRT-P2												X	
NFRT-P3													
NFRT-P4													X
NFRT-P5													
NFRT-P6	X												
NFRT-P7		X											
NFRT-P8			X										
NFRT-P9											X		
NFRT-P10				X									
NFRT-P11						X							
NFRT-P12							X						
NFRT-P13													
NFRT-P14													
NFRT-P15													
NFRT-P16								X					
NFRT-P17													
NFRT-P18	X												

Table 5: Traceability Between Nonfunctional Test Cases and Nonfunctional Requirements, P-RF5 to OE-P1

Test IDs	**Nonfunctional Requirement IDs**											
	P-RF5	**P-RF6**	**P-C1**	**P-C2**	**P-SE1**	**P-SE2**	**P-SE3**	**P-L1**	**P-L2**	**OE-EPE1**	**OE-IAS1**	**OE-P1**
NFRT-P3	X											
NFRT-P5		X										
NFRT-P13				X								
NFRT-P14								X	X			
NFRT-P15							X					
NFRT-P17			X		X							
NFRT-P18						X						
NFRT-OE1												X
NFRT-MS3										X	X	

Table 6: Traceability Between Nonfunctional Test Cases and Nonfunctional Requirements, MS-M1 to CUL-C2

Test IDs	**Nonfunctional Requirement IDs**										
	MS-M1	**MS-S1**	**MS-S2**	**MS-A1**	**S-A1**	**S-IG1**	**S-P1**	**S-P2**	**S-P3**	**CUL-C1**	**CUL-C2**
NFRT-MS1	X										
NFRT-MS2		X	X								
NFRT-MS3				X							
NFRT-S1					X						
NFRT-CUL1										X	X

5 Unit Test Description

This section includes all unit tests for functional and non-functional requirements. Though most of the requirements will be tested manually, unit testing will still be utilized for some basic functionalities of all components. Since the team has not completed a detailed design document, this section cannot refer to all modules in MIS. Instead it will refer to a list of core modules to be implemented:

- **M1** User
- **M2** Friend
- **M3** Building
- **M4** Event
- **M5** Lecture
- **M6** Database

It is the first draft of the implementation design, and a more detailed version of unit test will be added once the MIS is completed.

5.1 Unit Testing Scope

All modules defined above are within the testing scope. The scope of unit testing was limited to the components that could be automatically tested. The AR module is out of the scope since it is largely composed of a third-parity library, Vuforia. It is a sophisticated AR software development kit, so we assumed that Vuforia's logic is correct and therefore shall not be tested. Also, the database module is out of the scope of functional requirement tests because it is largely composed of third-party database. But there are some non-functional requirements that can be tested

5.2 Tests for Functional Requirements

The following sections details unit tests for functional requirements. It is an essential aspect of testing as it verifies whether the modules are behaving correctly given the requirements in the SRS.

5.2.1 User Module

This section will contain all unit tests for module "User". The tests are chosen based on common user flows and cover the most important methods of this module.

1. FRT-M1-1

   Name: Password Setting

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: There is an old password for an account

   Input: New password

   Output: The old password is replaced by the new one

   Test Case Derivation: Verify the password reset functionality works well

   How test will be performed: Create a test case in UTF that sets a new password for a user and verify the password has been changed

2. FRT-M1-2

   Name: Search For Valid Username

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: There are some valid users during the test

   Input: One of the valid usernames

   Output: The user is found from the list by its username

   Test Case Derivation: Verify that user can search for a user by username

   How test will be performed: Create a test case in UTF that searches for a existing username and verify the corresponding user is found

3. FRT-M1-3

   Name: Search For Invalid Username

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: There are some valid users during the test

Input: Invalid usernames

Output: No user is found

Test Case Derivation: Verify that user can search for a user by username

How test will be performed: Create a test case in UTF that searches for a non-existing username and verify no user is found

4. FRT-M1-4

   Name: Get Location

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: User agrees to share location

   Input: User request

   Output: Geographic position of the device with accuracy of GPS_ACCURACY

   Test Case Derivation: Verify that user can share their location accurately

   How test will be performed: Create a test case in UTF that searches for a non-existing username and verify no user is found

5.2.2 Friend Module

This section will contain all unit tests for module "Friend". The tests are chosen based on common user flows and cover the most important methods of this module.

1. FRT-M2-1

   Name: Send new message

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: There are two accounts being friends

   Input: A message sent friend to the friend

   Output: The Message is added to chat history

   Test Case Derivation: Verify that messages can be sent and stored between friends

   How test will be performed: Create a test case in UTF that messages a friend and verify the message is added to chat history

2. FRT-M2-2

   Name: Receive new message

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: There are two accounts being friends

   Input: A message sent from the friend

   Output: The Message is added to chat history

   Test Case Derivation: Verify that messages can be sent and stored between friends

   How test will be performed: Create a test case in UTF that receives a message from a friend and verify the message is added to chat history

5.2.3 Building Module

This section will contain all unit tests for module "Building". The tests are chosen based on common user flows and cover the most important methods of this module.

1. FRT-M3-1

   Name: List Events

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: There are some events for this target building

   Input: User request

   Output: All events in the building are listed

   Test Case Derivation: Verify that all events of the building are shown with nothing missing

   How test will be performed: Create a test case in UTF where a user shares the location, verify the location information retrieved matches the expected GPS coordinates

2. FRT-M3-2

   Name: List Lectures

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: There are some lectures for this target building

Input: User request

Output: All lectures in the building are listed

Test Case Derivation: Verify that all lectures of the building are shown with nothing missing

How test will be performed: Create a test case in UTF that has a building with a list of lectures, verify all lectures are displayed once lecture information is requested

3. FRT-M3-3

   Name: Check User Location in Bounds

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: Building has a bounding box outlined by GPS coordinates

   Input: User location

   Output: True if the user location is within the bounding box, False otherwise

   Test Case Derivation: Verify that the GPS building detection algorithm is correct

   How test will be performed: Create a test case in UTF that has a building object with a bounding box and a user object with a dummy location, verify that the algorithm identifies cases where the user is within the bounding box

5.2.4 Event Module

This section will contain all unit tests for module "Event". The tests are chosen based on common user flows and cover the most important methods of this module.

1. FRT-M4-1

   Name: Successful New Event Creation

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: User is logged in as admin

   Input: Event information including location, time and club/department

Output: Event is successfully added to the system

Test Case Derivation: Verify that new event can be added by an administrator only if all information is provided

How test will be performed: Create a test case in UTF that creates a new event with all information provided, verify the event is added to the event list

2. FRT-M4-2

   Name: Unsuccessful New Event Creation

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: User is logged in as admin

   Input: Event information with one of the following missing: location, time or club/department

   Output: Event cannot be added to the system, an error pops up saying information missing

   Test Case Derivation: Verify that new event can be added by an administrator only if all information is provided

   How test will be performed: Create a test case in UTF that creates a new event with some necessary information missing, verify the event fails to be added to the event list

5.2.5 Lecture Module

This section will contain all unit tests for module "Lecture". The tests are chosen based on common user flows and cover the most important methods of this module.

1. FRT-M5-1

   Name: Successful New Lecture Creation

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: User is logged in as admin

   Input: Lecture information including location, time and instructor

   Output: Lecture is successfully added to the system

Test Case Derivation: Verify that new lecture can be added by an administrator only if all information is provided

How test will be performed: Create a test case in UTF that creates a new lecture with all information provided, verify the lecture is added to the lecture list

2. FRT-M5-2

   Name: Unsuccessful New Lecture Creation

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: User is logged in as admin

   Input: Lecture information with one of the following missing: location, time or instructor

   Output: Lecture cannot be added to the system, an error pops up saying information missing

   Test Case Derivation: Verify that new lecture can be added by an administrator only if all information is provided

   How test will be performed: Create a test case in UTF that creates a new lecture with some necessary information missing, verify the lecture fails to be added to the lecture list

5.3 Tests for Nonfunctional Requirements

This application does not many non-functional requirements to be tested with unit testing. Most of non-functional requirements will be verified through system tests in section 4.2. The following section includes all non-functional requirements that are skipped in system test (like P-PA2, S-IG1, etc.).

5.3.1 Database Module

This section will contain all unit tests for module "Database". No functional requirements will be tested here since it is a third party component and all the methods can be trusted.

1. FRT-M6-1

   Name: Unique keys

Type: Functional, Dynamic, Automatic, Unit

Initial State: Some user exists in the module

Input: A new user to be added with same username

Output: An error appears saying the username already exists

Test Case Derivation: Verify that the system prevents creating account with sample username

How test will be performed: Create a test case in UTF that adds a new user to the database and verify the request is rejected

5.3.2 Database Module

This section will contain a unit tests for module "User". This test will verify the style test about color contrast. Since the module is mostly composed of a third party library, it is unnecessary to test all methods in functional requirement section.

1. FRT-M1-5

   Name: Color Contrast

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: User is logged in

   Input: The user change the page color

   Output: The colour contrast for each page is calculated to be at least 4.5:1

   Test Case Derivation: Verify that the page components are contrast enough for items on the screen to be more easily visible

   How test will be performed: Create a test case in UTF that takes all the possible color combinations on each page as an input and calculates the contrast, and checks if that contrast is at least 4.5:1

5.4 Traceability Between Test Cases and Modules

Table 7: Traceability Between Test Cases and Modules

Test IDs	Module IDs					
	M1	**M2**	**M3**	**M4**	**M5**	**M6**
FRT-M1-1	X					
FRT-M1-2	X					
FRT-M1-3	X					
FRT-M1-4	X					
FRT-M1-5	X					
FRT-M2-1		X				
FRT-M2-2		X				
FRT-M3-1			X			
FRT-M3-2			X			
FRT-M3-3			X			
FRT-M4-1				X		
FRT-M4-2				X		
FRT-M5-1					X	
FRT-M5-2					X	
FRT-M6-1						X

6 Nondynamic Test Plan

This section summaries all the nondynamic test the team will conduct in section 4 system testing and section 5 unit testing. Most of the tests will be in the form of code review, document review or code walkthroughs.

Table 8: Nondynamic Test Plan

Test Method	**Description**	**Related Requirement(s)**
Code Inspection for user info	It will be held when the implementation is done with the attendance of the supervisor. It will convince people that the data collection is totally legitimate	P-SC3
Database Walkthrough	A code walkthrough focusing on database connection and its storage. It will be held when the implementation is done and convince people that the database is reliable will not store information without user's permission	P-SC1, P-C2
Data Transmission Walkthrough	A code walkthrough focusing on data transmitting. It will be held when the implementation is done and convince people that the application transmit no data while no in use	P-SC4
Longevity Peer Evaluation	A peer evaluation session to convince people that the produce can operate for a long time. It will be held when the implementation is done by another developers' team. Though it does not 100% fulfill the requirements, it builds some confidence	P-L1, P-L2
Document Review	The peer evaluation happens after the deadline of all important documents. During the document review, the other team points out problems and improve document quality	All
Weekly Code Review	The developer will have weekly code review with the team for better code quality and formatting	All

7 Appendix

This is where you can place additional information.

7.1 Symbolic Parameters

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

Table 9: **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
GPS_ACCURACY	Accuracy of location sharing	25 meter
SCREEN_VIEWPORTS	List of all popular mobile screen sizes	360X740, 390X844, 820X1180
USER_PROFILE	All information about the user	username, password, gender, age, email, area of study

7.2 Usability Survey Questions

Please rate each of the following questions on a rating scale of 1-10 where 10 represents the most positive experience, and 0 illustrates the most negative experience.

1. **User-friendly Layout:** Do you think that the layout of this product is user-friendly?

2. **Intuitive icons:** Do you feel comfortable running different features without needing assistance or a tutorial?

3. **Immediate Visual Response when Clicking:** Does the application provide visual feedback when using different features/switching betewen pages?

4. **Appealing Colour Scheme:** Do you think that the colour scheme chosen for the interface is appealing?

5. **Easy to Navigate:** Do you feel that the user interface is easy to navigate?

6. **Helpful Tutorial:** After viewing the tutorial, do you feel that you have a better understanding of the product's features?

7. **No Technical or Software-specific Language:** Do you believe that the product avoids the use of technical/software-specific language unless necessary?

8. **Common periods of usage:** At what times of the day/week do you find yourself using the product?

9. **Cultural Friendliness:** Do you agree that the product does not display any offensive language/symbols?

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:

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?

- Static Testing Knowledge - Matthew Miller

  One skill that the team will need to complete the verification and validation of the project is static testing knowledge. One way to master this skill is by practicing through code reviews, which involve reading through the code to check for errors. This can be done with our own code or another group's code. Another approach to acquiring this knowledge is by watching tutorials on how to do static testing to help further understanding of the procedure.

  I plan to acquire knowledge of static testing to gain a better understanding of the procedure, as well as putting that knowledge into practice. I chose these approaches because I find that I learn best by applying knowledge in real-world settings.

- Stress Testing, Apache JMeter - Zihao Du

  One of the skill the team needs to acquire for testing purpose is stress testing since the application has a max capacity of users and the team need to check if the server is robust enough to fulfil the requirement. One approach to master this skill is to read their official user manual and tutorial – as an Apache project, it has a detailed tutorial on the website for beginners. Another approach to acquire the knowledge is

to watch some crach course videos about how to use JMeter for stress test like this.

I decide to learn this skill by reading the user manual on their website. It is not because JMeter has a detailed tutorial with step by step screenshot which is very easy to understand, but the official documents can give me a better understanding of the tool itself in general since we need it for both server and database load testing.

- Automated and Manual Dynamic Testing Knowledge - Michael Kim

  For automated dynamic testing knowledge, learning about tools that show code coverage will allow me to identify any aspects of the product that require further testing. I am interested in this as I had many instances where issues previously caught reoccured after a different fix because I had not automated the test cases and neglected them. This will help me set up automated tests anyone can run before pushing to GitHub to check that their code did not break a previously functional segment of the product. Another method of aquiring this skill would code review and documentation to understand what purpose each of the functions serve. This will help me write comprehensive tests and cover all known use cases of the code. I am interested in this method as the product may have other maintainers join in the future, and having these documentations and comments will help in their understanding of the product and continuing the proper maintenance.

  Another aspect of automated dynamic testing that I must learn is how to set up the automated tests and run them all For manual dynamic testing knowledge, I will learn about proper usage of Burp Suite, Android Studio Logging tools, Valgrind, and other monitoring and logging applications to help verify the activities of our product. I chose to learn more about this because my experiences in Co-Op as Security Specialist Intern piqued by interest in software testing and verification of security requirements. Through this experience, I have tested applications to see whether they adhere to security requirements imposed by the non-functional requirements. There are various tutorials that can be found on YouTube and extensive documentation provided by the community that uses these tools. This will help significantly in understanding the specific uses of these tools and allow me to test rapidly when a new prototype or update is built. Another approach in learning this technique

would be going through bug reports and reproduction steps outlined for other applications. From my Co-Op experience, I had to go through some of these reports so that our products would not experience the same failures other companies have. I am interested in learning more about this process and see what other issues were found and how they were fixed. This will help in grasping the proper procedure of manual dynamic testing and potential failures. There are detailed reports and research on existing products that has experienced significant failures, such as the 2022 Rogers Communications outage. This will also help in prioritization of issues so that critical fixes and failures are patched first.

- Testing Tools: Unity Test Framework - Waseef Nayeem

  One of the specific technologies that will be used for verification and validation is the Unity Test Framework. Unity provides a fully-featured solution for unit testing and also provides tools for automating more complex tests. There is extensive documentation on the UTF on Unity's website as well as several videos and examples that can be used to learn. Another way to learn and master is through experimentation. I could create some basic test assemblies and unit test and modify the inputs or scale up to more complex testable cases.

  Since my role in the project is focused on Unity development I will research and learn how to use the Unity Test Framework. I will accomplish this by quickly learning the basics of UTF and then experimenting early with test cases. I indicated in my previous reflection that software testing is an area I would like to improve in. This will allow me to develop better testing methodologies and improve the quality of future software projects.

- Object detection domain knowledge - Abhiram Neelamraju

  Knowledge in the field of object detection is something that could be of great use to the developers. It is something that could help the rest of the team along with developers figure out what is within and what is beyond the scope of the project. We can use this information to better plan our features and come up with a realistic expectation for our product. For example, we can research into Vuforia and what its

good at and what it might have problems with.

There are many ways to obtain this domain knowledge regarding object detection. There are several free courses available on websites such as coursera, udemy etc. covering relevant content. There are also youtube tutorials that cover similar content especially specific to Vuforia. My personal choice of learning method would be youtube. There are many youtube videos covering basic principles of object detection along with end-to-end project examples implementing Vuforia. This is a great efficient tool to learn from for free.

- Knowledge of AR Standards - Firas Elayan

It is essential that the team acquires knowledge of existing AR standards, as it will be needed to effectively verify and validate our application. These standards tell us about the quality that our product must have. They can also act as guidelines for creating a user interface that works well and is appealing to users. Knowledge of these standards and applying them to the verification and validation of our software will ensure it follows the best practices and performs at its highest level.

One way to learn this knowledge is to watch tutorial videos. Our supervisor has recommended some videos for beginners, and another way to acquire this domain knowledge is to read documentations of third party library, for example user manual of Unity 3D. I want to learn these standards by watching videos, these crash course videos can help me know the standards quickly and apply what I learned to ensure the quality of the application is the best it could be.