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

April 3, 2024

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
Feb 21	1.1	Revision 1: Introduce new usability testing
Feb 25	1.1	Revision 1: Resolve TA Feedback: add TA review in the plan and fix grammar inconsistency
Feb 27	1.1	Revision 1: Resolve TA Feedback for functional requirements: make functional tests more specific and add/remove tests according to Revision 1 requirements
Mar 2	1.1	Revision 1: Resolve TA Feedback for non-functional requirements: add usability survey content to non-functional tests
Mar 3	1.1	Revision 1: Add unit tests according to the design documentation; Add new non-dynamic tests to the table
Apr 3	1.1	Revision 1: Changes related to VnV Report feedback

Contents

List of Tables

1 Symbols, Abbreviations, and Acronyms

symbol	description
AR	Augmented Reality
Apache JMeter	A load testing tool for performance analysis
APK	Android Package, a file format used by Android
CAPTCHA	Human Verification Test
FEMA	Failure Mode and Effect Analysis
GPS	Global Positioning System
HA	Hazard Analysis
JHE	John Hodgins Engineering Building
Mapbox	A provider of custom online maps for websites and applications
MIS	Module Interface Specification
SRS	Software Requirement Specification
TA	Teaching Assistant
UI	User Interface
URI	Uniform Resource Identifier, identifies an abstract or physical resource
UTF	Unity Test Framework
VnV	Verification and Validation
Vuforia	AR software development kit

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 their corresponding methods tests. The team will iterate on the VnV plan and have a more detailed version with the test code when the modules are specified in the design document.

2 General Information

2.1 Summary

The software we test 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 and filter lectures and events
- Allowing the user to customize their schedule – pin and unpin events or lectures based on their interest.
- Identifying major campus buildings through the user's location and camera view, and using the information to display relevant lectures and events.

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.

- Advanced extensibility and maintenance requirements:

  This is because the project only aims to support senior software students for now and We are not expected to serve all students by the end of Revision 1. Maintenance requirements are designed for maintainers to maintain the project well in the future (after Revision 1 if the project can be used in real life), not developers. Therefore they will be out of the scope of this documentation.

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, Unit tester**, Leads the testing process, runs unit tests and generates test reports
Michael Kim	**Usability Tester**, Tests the user interface of the software to assess how intuitive the layout and UI 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 decided 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'll be 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.

- Ad-hoc review by TA:

  We will perform a review of our SRS with our TA, Sam. This meeting will be used to discuss how we can resolve Revision 0 feedback and we can also get some more suggestions about what can be improved in Revision 1.

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.

- TA review:

  We will conduct reviews with our TA Same to get his ideas about our designs. He can also help us resolve feedback he gave for Revision 0.

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.

- TA feedback:

  Our TA, Sam will give us feedback on VnV plan. This feedback will be useful to improve our documentation in Revision 1.

- 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 testing. 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 the 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 robustness and accuracy of each module when working individually. 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 techniques 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 dotnet format for C# to format our code. 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 in both Play Mode and Edit mode and there are other UI driven end to end automated tool that we can use if it is manual testing spends too much time.

For all tests or linter mentioned above, we will build a continuous integration pipeline in our Github repo so that they can be tested automatically.

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-Registration 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 create an account and use the application, the user must agree to the terms and conditions in the consent form. The following tests will have random accounts as input and we assume the success of the test on random accounts indicates the success of the test on all accounts.

1. **FRT-PR1**

   Name: Agree To Consent Form

   Control: Manual

   Initial State: The user does not have an account, and they starts to register an account. 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' and continues to complete the registration process

   Output: A notification shows the registration succeeds and 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 and register a new account. Then they will open up the consent form and click on 'Agree' button, try to finish the registration and verify the registration is successful and the login screen shows up as well

Related Requirement(s): FR-1-1

2. **FRT-PR2**

   Name: Disagree To Consent Form

   Control: Manual

   Initial State: The user does not have an account, and they starts to register an account. 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' and continues to complete the registration process

   Output: The registration fails and a warning will show up notifying the user that they cannot create an account unless they agree to the consent form

   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 and register a new account. Then they will open up the consent form and click on 'Disagree' button, try to finish the registration and verify a warning shows 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-2-6), along with user avatar settings (FR-2-5), password management(FR-2-4) and profile management (FR-2-7). User

accounts and virtual avatars are necessary for all users who want to use this application. The following tests will have random accounts as input and we assume the success of the test on random accounts indicates the success of the test on all accounts.

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:

   - Email: testUA1@gmail.com
   - password: FRT-UA1
   - nickname: UA1

   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 a test account with all information above and verify that the account can be logged into

   Related Requirement(s): FR-2-1

2. **FRT-UA2**

   Name: Unsuccessful Account Creation With Existing Email

   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:

   - Email: qtest@gmail.com (this is an existing test account)

- password: FRT-UA1
- nickname: UA1

Output: Account creation fails with a warning telling the user the email has already been used

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

How test will be performed: The tester will create an account with the information above and verify that the registration fails with a warning

Related Requirement(s): FR-2-1

3. **FRT-UA3**

   Name: Successful Account Login

   Control: Manual

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

   Input: All information needed to login:

   - Email: FRT-UA3@test.com (this account exists in the system already)
   - password: FRT-UA3

   Output: User successfully logs into the application and goes to the menu page

   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 login with the email and password given above and verifies that the login succeeds and the user is redirected to the menu page

   Related Requirement(s): FR-2-3

4. **FRT-UA4**

   Name: Unsuccessful Account Login With Wrong Password

   Control: Manual

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

   Input: All information needed to login:

   - Email: FRT-UA3@test.com (this account exists in the system already)
   - password: FRT321 (wrong password)

   Output: Login fails with a warning 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 not log in with a wrong password

   How test will be performed: The tester will login with the email and password given above and verifies that the login fails with a warning

   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

   - Email: FRT-UA5@gmail.com (this is an existing test account)
   - password: FRT-UA5
   - nickname: UA5

   Input: User clicks on the delete account button on the profile page and confirms the deletion

   Output: The user is redirected to the login page and the account cannot be logged in any more

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 account above and verify the account cannot be logged in any more

Related Requirement(s): FR-2-2

6. **FRT-UA6**

   Name: Reset Account Password

   Control: Manual

   Initial State: The user has an account:

   - Email: campusconnections@gmail.com (this is an existing test account)
   - password: qtesting

   Input: Email address and new password

   - new password: QTesting

   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 user will click on 'Forget Password' button on the login page and enter the email address above. Then the user will follow the instruction on the email sent by the system and enters a new password. After that the user will login with the new password to verify the new password works properly

   Related Requirement(s): FR-2-4

7. **FRT-UA7**

   Name: Avatar Creation and Modification

   Control: Manual

   Initial State: The user has an account with DEFAULT_AVATAR

   Input: URI represents the new avatar:

- URI: https://upload.wikimedia.org/wikipedia/commons/2/2f/Google_2015_logo.svg

Output: The user changes the avatar to a Google logo

Test Case Derivation: To help users enjoy social networking when using this application, we should allow them to customize 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 edit the avatar URI and change it to be the URI above. Then the tester will verify the avatar has been changed to a Google logo

Related Requirement(s): FR-2-5

8. **FRT-UA8**

   Name: Email Verification

   Control: Manual

   Initial State: The user has an account whose email has not been verified yet

   - Email: fuz15@mcmaster.ca (this is an existing test account)
   - password: password

   Input: User clicks on 'Verify Email' button on user profile page and follows instructions on the email sent from the system

   Output: That email above is verified as a valid email address

   Test Case Derivation: For students to get access to events and lectures, they must prove the system that they have a valid school email, therefore, email verification is important.

   How test will be performed: The tester will login with the email and password given above and go to user profile page and click on 'Verify Email' button. Then the tester will follow the instructions of the email sent by the system to verify the email. Finally the tester will login and verify that the email verification button is disabled because the email has been verified

   Related Requirement(s): FR-2-6

9. **FRT-UA9**

   Name: Edit Profile

   Control: Manual

   Initial State: The user has an account

   - Email: qtest@gmail.com (this is an existing test account)
   - password: qtesting
   - program: Software Engineering

   Input: New Profile:

   - newProgram: Computer Science
   - newLevel: 4

   Output: The program and level are updated

   Test Case Derivation: To help users enjoy social networking when using the application, we should allow them to edit their information whenever they want

   How test will be performed: The tester will login with the email and password given above and go to user profile page and click on 'Edit' button. Then the tester will enter new program and level and save the changes Finally the tester will verify the profile has been updated

   Related Requirement(s): 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. The following tests will have random accounts as input and we assume the success of the test on random accounts indicates the success of the test on all accounts.

1. **FRT-SN1**

   Name: Successful Friend Request

   Control: Manual

   Initial State: The user is logged in with the following account:

   - Semail: FRT-SN1@test.com
   - password: testing

   Input: A valid email to send the request:

   - Temail: FRT-SN1-F@test.com

   Output: A Request is sent to the target user

   Test Case Derivation: Users should be able to make friends on this social media platform by searching for their ID (email) and sending requests

   How test will be performed: The tester will login and send a friend request from the account given in initial state to the target user. Then the tester will login to the target account and verify that there is a pending friend request

   Related Requirement(s): FR-3-1

2. **FRT-SN2**

   Name: Friend Request Acceptance

   Control: Manual

   Initial State: A friend request was sent from an account (Semail) to the target account (Temail):

   - Semail: FRT-SN1@test.com
   - Temail: FRT-SN1-F@test.com

   Input: The request is accepted

   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 login to target account and accept the request. Then the tester will login to sender account to verify that they are in friend lists of each other

Related Requirement(s): FR-3-1, FR-3-6

3. **FRT-SN3**

   Name: Friend Request Rejection

   Control: Manual

   Initial State: A friend request was sent from an account (Semail) to the target account (Temail):

   - Semail: FRT-SN1@test.com
   - Temail: FRT-SN1-F@test.com

   Input: The request is rejected

   Output: The request is declined and no friend is added for both accounts

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

   How test will be performed: The tester will first login to target account and reject the request. Then the tester will login to sender account to verify that they are not in friend lists of each other

   Related Requirement(s): FR-3-1, FR-3-6

4. **FRT-SN4**

   Name: Friend Deletion

   Control: Manual

   Initial State: A friend (Femail) exist in the friend list of the test account (Temail):

   - Temail: FRT-SN4@test.com
   - Femail: FRT-SN4-F@test.com

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 login to the test account (Temail) and delete the friend (Femail) 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 (Femail) exist in the friend list of the test account (Temail):

   - Temail: FRT-SN5@test.com
   - Femail: FRT-SN5-F@test.com

   Input: Message: 'Hello World'

   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 log in to both the test account and the friend account. Then the tester will open the chat and send the input message from test account and verify that message shows up in the friend account chat

   Related Requirement(s): FR-3-3

6. **FRT-SN6**

   Name: Friend Sharing Event

   Control: Manual

   Initial State: A friend (Femail) exist in the friend list of the test account (Temail):

- Temail: FRT-SN6@test.com
- Femail: FRT-SN6-F@test.com

Input: Message that contains event name and follows some specific pattern: Hey, check this event: _E_[EXPO]

Output: User will be redirect to the event page with that event once they click on the message

Test Case Derivation: Users shall be able to share events they found with friends conveniently so that they can join events together in person.

How test will be performed: The tester will log in to both the test account and the friend account. Then the tester will open the chat and send that message from test account and verify that message can be clicked in the friend account chat and it redirects to event page correctly.

Related Requirement(s): FR-3-4

7. **FRT-SN7**

   Name: Friend Sharing Lecture

   Control: Manual

   Initial State: A friend (Femail) exist in the friend list of the test account (Temail):

   - Temail: FRT-SN7@test.com
   - Femail: FRT-SN7-F@test.com

   Input: Message that contains lecture code and follows some specific pattern: 'Hey, are you in this lecture: _L_[SFRWENG 4G06]'

   Output: User will be redirect to the lecture page with that lecture once they click on the message

   Test Case Derivation: Users shall be able to share lectures they found with friends conveniently so that they can meet more classmate and study together.

   How test will be performed: The tester will log in to both the test account and the friend account. Then the tester will open the chat

and send that message from test account and verify that message can be clicked in the friend account chat and it redirects to lecture page correctly.

Related Requirement(s): FR-3-5

4.1.4 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. The test cases ensure that the lectures and events information is accurate and helpful for all users. The following tests will have random lecture/event as input and we assume the success of the test on random event/lecture indicates the success of the test on all of them.

1. **FRT-LE1**

   Name: Save Event

   Control: Manual

   Initial State: A sample event:

   - Name: EXPO

   Input: On the event page, user clicks on the save button on the pop-up window with details of the sample event

   Output: The event is saved to the user's event list

   Test Case Derivation: The user should be able to interact with events available on campus and customize their own event list. Therefore, they should be able to save events for later

   How test will be performed: The tester will go to event page and find the sample event, click on the save button on the pop-up window. Then the tester will verify the event shows up in the user's saved event list.

   Related Requirement(s): FR-4-1

2. **FRT-LE2**

   Name: Unsave Event

Control: Manual

Initial State: A sample event that is already been saved:

- Name: EXPO

Input: On the event page, user clicks on the unsave button on the pop-up window with details of the sample event

Output: The event is removed from the user's event list

Test Case Derivation: The user should be able to interact with events available on campus and customize their own event list. Therefore, they should be able to unsave events if that is no longer needed

How test will be performed: The tester will go to event page and find the sample event, click on the unsave button on the pop-up window. Then the tester will verify the event disappears in the user's saved event list.

Related Requirement(s): FR-4-1

3. **FRT-LE3**

Name: Save Lecture

Control: Manual

Initial State: A sample lecture:

- Code: SFWRENG 4G06

Input: On the lecture page, user clicks on the save button on the pop-up window with details of the sample lecture

Output: The event is saved to the user's lecture list

Test Case Derivation: The user should be able to interact with lectures available on campus and customize their own lecture list. Therefore, they should be able to save lectures for later

How test will be performed: The tester will go to lecture page and find the sample lecture, click on the save button on the pop-up window. Then the tester will verify the event shows up in the user's saved lecture list.

Related Requirement(s): FR-4-2

4. **FRT-LE4**

   Name: Unsave Lecture

   Control: Manual

   Initial State: A sample lecture that is already been saved:

   - Code: SFWRENG 4G06

   Input: On the lecture page, user clicks on the unsave button on the pop-up window with details of the sample lecture

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

   Test Case Derivation: The user should be able to interact with lectures available on campus and customize their own lecture list. Therefore, they should be able to unsave lectures if that is no longer needed

   How test will be performed: The tester will go to lecture page and find the sample lecture, click on the unsave button on the pop-up window. Then the tester will verify the event disappears in the user's saved lecture list.

   Related Requirement(s): FR-4-2

5. **FRT-LE5**

   Name: Administrator Add Event

   Control: Manual

   Initial State: User is logged in as an administrator

   - email: campusconnections@gmail.com
   - password: testing

   Input: Sample event:

   - name: Test event
   - description: Sample event for system test
   - time: 0
   - duration: 0
   - location: Online

- isPublic: true
- organizer: Team 2

Output: The event is added to the event list

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 is shown in the list with all information

Related Requirement(s): FR-4-3

6. **FRT-LE6**

   Name: Administrator Add Event With Existing Name

   Control: Manual

   Initial State: User is logged in as an administrator

   - email: campusconnections@gmail.com
   - password: testing

   An event with following information exists:

   - name: Test event
   - description: Sample event for system test
   - time: 0
   - duration: 0
   - location: Online
   - isPublic: true
   - organizer: Team 2

   Input: New sample event:

   - name: Test event
   - description: New Description
   - time: 100

- duration: 30
- location: NA
- isPublic: false
- organizer: NA

Output: The sample event information is changed to the one in the input

Test Case Derivation: Administrators know the event name should be unique and it is unlikely that they give a different event the same name, so this happens most likely when they want to update an event (for instance monthly updated club event)

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 is still in the list with all information updated

Related Requirement(s): FR-4-3

7. **FRT-LE7**

   Name: Administrator Edit Event

   Control: Manual

   Initial State: User is logged in as an administrator

   - email: campusconnections@gmail.com
   - password: testing

   Input: Sample event name and new location:

   - name: Test event
   - location: ITB AB102

   Output: The test event location is updated 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 if that is out of date

   How test will be performed: The tester will log in as an administrator, find the sample event and update the sample event location. Then the tester will verify the event location shown in the list is updated

Related Requirement(s): FR-4-3

8. **FRT-LE8**

   Name: Administrator Delete Event

   Control: Manual

   Initial State: User is logged in as an administrator

   - email: campusconnections@gmail.com
   - password: testing

   Input: Sample event (already in the system) name:

   - name: Test event

   Output: The event is deleted and disappears from the list

   Test Case Derivation: Administrators work as a source of truth in this application, therefore they shall be able to clean old events

   How test will be performed: The tester will log in as an administrator, find the sample event and click on delete button. Then the tester will verify the event disappears from the list

   Related Requirement(s): FR-4-3

9. **FRT-LE9**

   Name: Administrator Add Lecture

   Control: Manual

   Initial State: User is logged in as an administrator

   - email: campusconnections@gmail.com
   - password: testing

   Input: Sample lecture:

   - code: TEST 1T03
   - name: Test lecture
   - time: 12:00 - 13:00, Mon
   - location: Online

- instructor: NA

Output: The lecture is added to the lecture list

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

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 is shown in the list with all information

Related Requirement(s): FR-4-4

10. **FRT-LE10**

Name: Administrator Add Lecture With Existing Code

Control: Manual

Initial State: User is logged in as an administrator

- email: campusconnections@gmail.com
- password: testing

A lecture with following information exists:

- code: TEST 1T03
- name: Test lecture
- time: 12:00 - 13:00, Mon
- location: Online
- instructor: NA

Input: New sample lecture:

- code: TEST 1T03
- name: New Test lecture
- time: 12:00 - 13:00, Wed
- location: TBD
- instructor: TBD

Output: The sample lecture information is changed to the one in the input

Test Case Derivation: Administrators know that lecture code should be unique and it is very unlikely that they give a different lecture the same code, so this happens most likely when they want to update a lecture (for instance in a new semester the lecture has a new schedule)

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 is still in the list with all information updated

Related Requirement(s): FR-4-4

11. **FRT-LE11**

    Name: Administrator Edit Lecture

    Control: Manual

    Initial State: User is logged in as an administrator

    - email: campusconnections@gmail.com
    - password: testing

    Input: Sample lecture name and new location:

    - code: TEST 1T03
    - location: ITB AB102

    Output: The test lecture location is updated 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 if that is out of date

    How test will be performed: The tester will log in as an administrator, find the sample lecture and update the sample lecture location. Then the tester will verify the lecture location shown in the list is updated

    Related Requirement(s): FR-4-4

12. **FRT-LE12**

    Name: Administrator Delete Lecture

    Control: Manual

    Initial State: User is logged in as an administrator

    - email: campusconnections@gmail.com
    - password: testing

    Input: Sample lecture (already in the system) name:

    - code: TEST 1T03

    Output: The lecture is deleted and disappears from the list

    Test Case Derivation: Administrators work as a source of truth in this application, therefore they shall be able to clean old lectures

    How test will be performed: The tester will log in as an administrator, find the sample lecture and click on delete button. Then the tester will verify the lecture disappears from the list

    Related Requirement(s): FR-4-4

13. **FRT-LE13**

    Name: Event Information

    Control: Manual

    Initial State: A sample event exists:

    - name: EXPO

    Input: User clicks on the sample event

    Output: All event information is shown in a pop-up window

    Test Case Derivation: All events should follow some pattern and have following information:

    - organizer
    - location
    - time

- duration

How test will be performed: The tester will find the sample event in the table and click on it. Then the tester will verify all entries listed above are shown in the pop-up window

Related Requirement(s): FR-4-5

14. **FRT-LE14**

Name: Lecture Information

Control: Manual

Initial State: A sample lecture exists:

- code: SFWRENG 4G06

Input: User clicks on the sample lecture

Output: All lecture information is shown in a pop-up window

Test Case Derivation: All events should follow some pattern and have following information:

- instructor
- location
- time
- code

How test will be performed: The tester will find the sample lecture in the table and click on it. Then the tester will verify all entries listed above are shown in the pop-up window

Related Requirement(s): FR-4-6

15. **FRT-LE15**

Name: Lecture Filter by code

Control: Manual

Initial State: Some software engineering lecture exists:

- SFWRENG 4G06
- SFWRENG 4E03

Input: Filter:

- FilterString: SFWRENG

Output: All lectures which does not contain the FilterString in the code are removed from the list

Test Case Derivation: In order to find lectures easily, filter by lecture course code is needed if the lecture list is very long

How test will be performed: The tester will find go to the lecture page and enter the FilterString and click the search button. Then the tester will verify only software engineering courses are left in the list

Related Requirement(s): FR-4-7

16. **FRT-LE16**

Name: Event Filter by Name

Control: Manual

Initial State: Some job fair event exists:

- Job Fair: March 4
- Job Fair: March 10

Input: Filter:

- FilterString: Job Fair

Output: All event which does not contain the FilterString in the name are removed from the list

Test Case Derivation: In order to find event easily, filter by event name is needed if the event list is very long

How test will be performed: The tester will find go to the event page and enter the FilterString and click the search button. Then the tester will verify only job fair events are left in the list

Related Requirement(s): FR-4-8

4.1.5 AR Camera

This section includes all test cases related to campus navigation with AR technology. This will include the requirements for building recognition (FR-5-1) and demonstrating activities in this building (FR-5-2). Augmented reality provides an immersive user experience and it is the unique selling point of the application. The following tests are about one of the buildings (JHE) only we assume the success of these tests indicates the success of the test on all other target buildings.

1. **FRT-AR1**

   Name: Successful Building Recognition

   Control: Manual

   Initial State: User is at the front door of JHE

   Input: Clear camera view

   Output: The building is recognized with name and description shown as an AR object

   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 to the front door of JHE, open AR camera and look into it, verifying the building is recognized and a name and description is displayed on the screen

   Related Requirement(s): FR-5-1

2. **FRT-AR2**

   Name: Unsuccessful Building Recognition

   Control: Manual

   Initial State: User is out of campus

   Input: Clear camera view

   Output: No AR objects is shown

   Test Case Derivation: Building recognition is the essential functionality for AR technology applied in this application. Any buildings out

of campus should not be recognized because they are out of the scope of this project and buildings that are not targets should not be recognized for high accuracy oft AR camera

How test will be performed: The tester will walk out of campus, open AR camera and look into it, verifying there is no objects shown on the screen

Related Requirement(s): FR-5-1

3. **FRT-AR3**

   Name: Building Lectures/Events

   Control: Manual

   Initial State: User is in JHE lobby

   Input: Clear camera view

   Output: Event and lecture information separated by room number at the corresponding locations of the building

   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 in JHE lobby, open AR camera and look into it, verifying there are lectures and events information shown at the rooms where they will be held

   Related Requirement(s): FR-5-2

4.1.6 Map and Location

1. **FRT-MAP1**

   Name: User Location

   Control: Manual

   Initial State: User allows the user to use their real-time location

   Input: User enters the map page

   Output: A model represents the user shows up on the map and moves correspondingly when the user moves

Test Case Derivation: For those who come to the campus for the first time, a map is necessary for them to find the building they want to go to

How test will be performed: The tester will go to map page and walk around on campus and verify if the model moves correspondingly on the map

Related Requirement(s): FR-6-1

2. **FRT-MAP2**

   Name: Friend Locations

   Control: Manual

   Initial State: User has some friends who are willing to share locations:

   - email1: MAP2-1@test.com
   - email2: MAP2-2@test.com

   Input: User enters the map page

   Output: Another models represent friends shows up on the map and moves correspondingly when friends moves

   Test Case Derivation: It will be fun to know where your friends are currently on campus and it makes it easier to have in person communications

   How test will be performed: Testers will login with friends account (email1 and email2), go to map page and walk around on campus and the other tester will verify if their model moves correspondingly on the map

   Related Requirement(s): FR-6-1

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: Survey taker is given an account:

   - email: mtest@gmail.com
   - password: mtesting

   Input/Condition: User follows the instructions and attempts to finish tasks in the survey

   Output/Result: The users should be able to complete tasks without any help and the average individual score is over MIN_SCORE in the"Immediate Visual Response when Clicking" and "Appealing Colour Scheme" categories of the 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 McMaster students

   Related Requirement(s): LF-A1, LF-S2

2. **NFRT-LF2**

   Name: Visual inspection with different screen sizes

   Type: Non-functional, Dynamic, Manual

   Initial State: NA

   Input/Condition: User opens the application on the phone with all different screen sizes in the SCREEN_VIEWPORTS list

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-A2

3. **NFRT-LF3**

   Name: Visual inspection of color scheme

   Type: Non-functional, Dynamic, Manual

   Initial State: NA

   Input/Condition: User opens the application on the phone

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

   How test will be performed: The tester 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 understandability and easy of use

   Type: Non-functional, Dynamic, Manual

   Initial State: Survey taker is given an account:

   - email: mtest@gmail.com
   - password: mtesting

   Input/Condition: User follows the instructions and attempts to finish tasks in the survey

   Output/Result: The user can finish tasks without help and the average individual score is over MIN_SCORE in the "No Technical or Software-Specific Language" category of the 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 McMaster students

Related Requirement(s): UH-EOU1, UH-UP1

2. **NFRT-UH2**

   Name: Walkthrough of user guide

   Type: Non-functional, Static, Manual

   Input/Condition: GitHub web page

   Output/Result: Convinced the participants that the main features are explained in the GitHub repo

   How test will be performed: The tester will use a guest GitHub account and navigate to the GitHub page and verify there exists explanation of application features

   Related Requirement(s): UH-L1

3. **NFRT-UH3**

   Name: Visual inspection of color contrast

   Type: Non-functional, Manual

   Initial State: NA

   Input/Condition: User opens the application checks the color contrast statically

   Output/Result: The color contrast is greater than 4.5:1, the Web Content Accessibility Guidelines' AA standards for accessibility

   How test will be performed: The tester will statically check if the color contrast is enough for items to be easily visible. (Contrast between background color and text)

   Related Requirement(s): UH-A1

4.2.3 Performance

1. **NFRT-P1**

   Name: AR camera recognition

Type: Non-functional, Dynamic, Manual

Initial State: The user is near a target building (JHE)

Input/Condition: User turns on AR camera

Output/Result: Corresponding AR objects appears quickly

How test will be performed: The tester will turn on the AR camera and look at JHE and verify that the recognition time ls less than RECOGNITION_TIME

Related Requirement(s): P-SL1

2. **NFRT-P2**

   Name: Real-time location update

   Type: Non-functional, Dynamic, Manual

   Initial State: User allows the application to use device location

   Input/Condition: User turns on the map and walks around on campus

   Output/Result: The user model on the map is updated when the user is moving

   How test will be performed: The tester will open the map and walk around, and verify that the model on the map is updated within LOCATION_UPDATE_TIME

   Related Requirement(s): P-SL2

3. **NFRT-P3**

   Name: AR camera accuracy

   Type: Non-functional, Dynamic, Manual

   Initial State: User is near a target building (JHE)

   Input/Condition: User turns on the AR Camera

   Output/Result: AR objects shows up

   How test will be performed: The tester will try the camera from different angles and verify that the success rate is at lease AR_ACCURACY

   Related Requirement(s): P-PA1

4. **NFRT-P4**

   Name: Warning 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: There is a pop-up window telling the user the internet is lost

   How test will be performed: The tester will open the application with offline mode and visually check the warning message appears

   Related Requirement(s): P-RF1

5. **NFRT-P5**

   Name: Rudimentary functions when the server connection is lost

   Type: Non-functional, Dynamic, Manual

   Initial State: Server is turned down

   Input/Condition: User opens the application

   Output/Result: The application still works with limited functionalities

   How test will be performed: The tester will open the application without an available server and check the map can still show the user location, the saved events and lectures and other user profile information can be viewed and changed

   Related Requirement(s): P-RF2

6. **NFRT-P6**

   Name: Code inspection for server restart

   Type: Non-functional, Static, Manual

   Input/Condition: Server settings

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

   - Server attempts to restart when it crashes

How test will be performed: A code inspection will be held with superior's attendance to show that the server is set to restart upon crash

Related Requirement(s): P-RF3

7. **NFRT-P7**

   Name: AR camera help button

   Type: Non-functional, Dynamic, Manual

   Initial State: User turns on the AR camera

   Input/Condition: User clicks the help button

   Output/Result: A message telling the user things that may affect AR camera appears

   How test will be performed: The tester will turn on the AR camera and click on the help button and visually check if the message shows up

   Related Requirement(s): P-RF4

8. **NFRT-P8**

   Name: Load testing for server

   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: The tester will use Apache JMeter for the load test simulation

   Related Requirement(s): P-SE1

9. **NFRT-P9**

   Name: Code inspection for database capacity

   Type: Non-functional, Static, Automatic

   Input/Condition: Database documentation

Output/Result: Successfully convinced the participants the following:

- The database has enough space to store all the user, lecture and event information

How test will be performed: A code inspection will be held with superior's attendance to show that the Firebase database official document states there will be enough capacity with the current plan

Related Requirement(s): P-SE2

10. **NFRT-P10**

    Name: Code walkthrough for adding new building

    Type: Non-functional, Static, Automatic

    Input/Condition: Source code

    Output/Result: Successfully convinced the participants the following:

    - A new target building can be added without causing the application running any slower

    How test will be performed: A code walkthrough will be held with the superior's attendance to show that an extra target building will not affect the performance of the application

    Related Requirement(s): P-SE3

11. **NFRT-P11**

    Name: Code Peer Evaluation For Longevity

    Type: Non-functional, Static, Manual

    Input/Condition: Source code

    Output/Result: Successfully convinced the participants 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 tester 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

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 wants to download the application

   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-EPE1, OE-P1

4.2.5 Maintainability and Support

1. **NFRT-MS1**

   Name: Survey for maintenance time

   Type: Non-functional, Static, Manual

   Initial State: Survey taker is given an account:

   - email: mtest@gmail.com
   - password: mtesting

   Input/Condition: User answers the question about "Common periods of usage"

   Output/Result: The average individual score is over MIN_SCORE for this question

How test will be performed: A survey will be given to at least SURVEY_SAMPLE_SIZE users where they will tell if they are most likely to use the application at school time only

Related Requirement(s): MS-M1

2. **NFRT-MS2**

   Name: Check for feature request

   Type: Non-functional, Dynamic, Manual

   Initial State: A public GitHub repo exists for this application

   Input/Condition: User goes to the GitHub repo

   Output/Result: User can read issues created by the team and also create new issues

   How test will be performed: The tester will verify the team's feature issues are open to view and new issues can be raised 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: NA

   Input/Condition: The application is installed on devices with Android 11 and above version

   Output/Result: The application works without any error about compatibility

   How test will be performed: The tester 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 over different Android versions.

   Related Requirement(s): MS-A1

4.2.6 Security

1. **NFRT-S1**

   Name: Access test

   Type: Non-functional, Dynamic, Manual

   Initial State: Three accounts with different accesses is ready:

   - Admin: campusconnections@gmail.com
   - User: mtest@gmail.com
   - Guest gtest@gmail.com

   Input/Condition: The user starts the application with the three accounts

   Output/Result: At each level of access, the application constrains the possible actions to what is specified in requirement S-A1, S-A2, S-A3.

   How test will be performed:

   (a) The tester will begin at the first level of access, which should only allow the user to view public events, account recovery page, and single single-user map

   (b) The tester will then test with the second level of access which should allow the user to do everything except edit, adding, or deleting activities

   (c) 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

   Related Requirement(s): S-A1, S-A2, S-A3

2. **NFRT-S2**

   Name: Special character warning

   Type: Non-functional, Dynamic, Manual

   Initial State: User starts to register

   Input/Condition: User enters nickname with some special character: '; DELETE *' (See all SPECIAL_CHAR in the appendix)

Output/Result: A warning message is displayed telling the user that the special characters are not allowed and stops the user from registering

How test will be performed: The tester will start to register and enter the nickname above and visually inspect if the warning message appears

Related Requirement(s): P-SC5

3. **NFRT-S3**

   Name: Email format warning

   Type: Non-functional, Dynamic, Manual

   Initial State: User starts to register

   Input/Condition: User enters a not email string in the email field: 'SELECT * FROM TABLE'

   Output/Result: A warning message is displayed telling the user that the input is not an email address and stops the user from registering

   How test will be performed: The tester will start to register and enters a email 'SELECT * FROM TABLE' and visually inspect if the warning message appears

   Related Requirement(s): P-SC6

4.2.7 Privacy

1. **NFRT-PVC1**

   Name: Code inspection for legitimate use of personal data

   Type: Non-functional, Static, Manual

   Input/Condition: Source code

   Output/Result: Successfully convinced the participants the following:

   - The usage of a user's personal 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): S-P1

2. **NFRT-PVC2**

   Name: Code inspection for removing unused accounts

   Type: Non-functional, Static, Manual

   Input/Condition: Source code

   Output/Result: Successfully convinced the participants the following:

   - Any accounts that are not active for a long time (a semester) will be removed from the system

   How test will be performed: A code inspection will be held with supervisor's attendance to show the system is set to clean inactive users periodically

   Related Requirement(s): S-P2

3. **NFRT-PVC3**

   Name: Code Walkthrough For User Personal Data

   Type: Non-functional, Static, Manual

   Input/Condition: Source code

   Output/Result: Successfully convinced the participants 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

4. **NFRT-PVC4**

   Name: Code Walkthrough For Data Transmission Encryption

   Type: Non-functional, Static, Manual

   Input/Condition: Source code

   Output/Result: Successfully convinced the participants the following:

   - The product only transmits encrypted data from server to user

   How test will be performed: A code walkthough will be held by the developers to show that the application transmit encrypted data only

   Related Requirement(s): P-SC2

5. **NFRT-PVC5**

   Name: Leaving campus warning

   Type: Non-functional, Dynamic, Manual

   Initial State: User opens the map

   Input/Condition: User moves out of the campus

   Output/Result: A warning message is displayed telling the user that the map is not available out of campus

   How test will be performed: The tester will open the map and walk out of campus and visually inspect if the warning message appears

   Related Requirement(s): P-SC4

4.2.8 User Safety

1. **NFRT-US1**

   Name: Warning when starting AR camera

   Type: Non-functional, Dynamic, Manual

   Initial State: User allows the application to use camera

   Input/Condition: User turns on the AR camera

   Output/Result: A warning telling the user to be aware of their surroundings is displayed upon start-up of the camera

How test will be performed: The testers will open the AR camera on one device and visually inspect if the message is shown

Related Requirement(s): P-SC3

4.2.9 Cultural

1. **NFRT-CUL1**

   Name: Survey for feedback on cultural requirements

   Type: Non-functional, Dynamic, Manual

   Initial State: Use has an account:

   - email: mtest@gmail.com
   - password: mtesting

   Input/Condition: User follows the instructions and attempts to finish tasks in the survey

   Output/Result: The average individual score is over MIN_SCORE in the "Cultural Friendliness" category of the Usability 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 McMaster students

   Related Requirement(s): CUL-C1

4.2.10 Compliance

1. **NFRT-COM1**

   Name: Code walkthrough on compliance requirements

   Type: Non-functional, Static, Manual

   Input/Condition: Source code

   Output/Result: Successfully convinced the participants the following:

   - The data collected will be handled as per the same legal requirements for the university

- The application can abide by the guidelines set by university staff

How test will be performed: A code walkthrough will be held with the supervisor's attendance to show that the project fulfils the compliance requirements

Related Requirement(s): COM-SC1, COM-SC2

4.3 Traceability Between Test Cases and Requirements

Table 1: Traceability Between Functional Test Cases and Functional Requirements, FR-1 to FR-3-5

Test IDs	**Functional Requirement IDs**												
	FR1-1	**FR2-1**	**FR2-2**	**FR2-3**	**FR2-4**	**FR2-5**	**FR2-6**	**FR2-7**	**FR3-1**	**FR3-2**	**FR3-3**	**FR3-4**	**FR3-5**
FRT-PR1	X												
FRT-PR2	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						
FRT-UA9								X					
FRT-SN1									X				
FRT-SN2									X				
FRT-SN3									X				
FRT-SN4										X			
FRT-SN5											X		
FRT-SN6												X	
FRT-SN7													X

Table 2: Functional Test Case and Requirement Traceability Cont.

Test IDs	**Functional Requirement IDs**											
	FR3-6	**FR4-1**	**FR4-2**	**FR-4-3**	**FR4-4**	**FR4-5**	**FR4-6**	**FR4-7**	**FR4-8**	**FR5-1**	**FR5-2**	**FR6-1**
FRT-SN2	X											
FRT-SN3	X											
FRT-LE1		X										
FRT-LE2		X										
FRT-LE3			X									
FRT-LE4			X									
FRT-LE5				X								
FRT-LE6				X								
FRT-LE7				X								
FRT-LE8				X								
FRT-LE9					X							
FRT-LE10					X							
FRT-LE11					X							
FRT-LE12					X							
FRT-LE13						X						
FRT-LE14							X					
FRT-LE15								X				
FRT-LE16									X			
FRT-AR1										X		
FRT-AR2										X		
FRT-AR3											X	
FRT-MAP1												X
FRT-MAP2												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-S1**	**LF-S2**	**UH-EOU1**	**UH-L1**	**UH-UP1**	**UH-A1**
NFRT-LF1	X			X				
NFRT-LF2		X						
NFRT-LF3			X					
NFRT-UH1					X		X	
NFRT-UH2						X		
NFRT-UH3								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-SC1**	**P-SC2**	**P-SC3**	**P-SC4**	**P-SC5**	**P-SC6**	**P-PA1**	**P-RF1**	**P-RF2**	**P-RF3**	**P-RF4**
NFRT-P1	X												
NFRT-P2		X											
NFRT-~~P3~~ PVC3			X										
NFRT-~~P4~~ PVC4				X									
NFRT-~~P5~~ US1					X								
NFRT-~~P6~~ PVC5						X							
NFRT-~~P7~~ S2							X						
NFRT-~~P8~~ S3								X					
NFRT-P3									X				
NFRT-P4										X			
NFRT-P5											X		
NFRT-P6												X	
NFRT-P7													X

Table 5: Traceability Between Nonfunctional Test Cases and Nonfunctional Requirements, P-SE1 to MS-A1

Test IDs	**Nonfunctional Requirement IDs**										
	P-SE1	**P-SE2**	**P-SE3**	**P-L1**	**P-L2**	**OE-EPE1**	**OE-P1**	**MS-M1**	**MS-S1**	**MS-S2**	**MS-A1**
NFRT-P8	X										
NFRT-P9		X									
NFRT-P10			X								
NFRT-P11				X	X						
NFRT-OE1						X	X				
NFRT-MS1								X			
NFRT-MS2									X	X	
NFRT-MS3											X

Table 6: Traceability Between Nonfunctional Test Cases and Nonfunctional Requirements, S-A1 to COMP-SC1

Test IDs	**Nonfunctional Requirement IDs**							
	S-A1	**S-A2**	**S-A3**	**S-P1**	**S-P2**	**CUL-C1**	**COMP-L1**	**COMP-SC1**
NFRT-S1	X	X	X					
NFRT-PVC1				X				
NFRT-PVC2					X			
NFRT-CUL1						X		
NFRT-COM1							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 some low level components. According to the module hierarchy in MG, the following modules will be tested with unit tests:

- **M1** User
- **M2** Event
- **M3** Lecture
- **M4** Pagination and Filter
- **M5** Authentication
- **M6** Database

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. Third-party library modules like Vuforia and Mapbox are out of the scope of unit testing and we will trust they are bug-free. Also, high-level modules in the hierarchy are usually responsible for rendering UI and reacting to user actions (e.g. user profile module, lecture list module) therefore they are out of the scope of unit testing as well (in fact they are not quite testable with Unity test framework). Instead, they are tested with manual testing.

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 the module "User". The tests are chosen based on common user flows and cover the most important methods of this module. The getter and setter methods are part of the unit test as well, but they will not be included in the section because they are trivial.

1. FRT-M1-1

   Name: Successful default user creation

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: NA

   Input: User identifier, email: unitTest@test.com

   Output: A user is successfully created with following attributes:

   - email: unitTest@test.com
   - nickname: ''
   - photoURI: null
   - program: ''
   - level: 0
   - lectures: empty list of string
   - event: empty list of string
   - friends: empty list of string
   - friendInvitation: empty list of string

   Test Case Derivation: Verify that a new user can be created by just specifying the email and the rest attributes will be filled automatically

   How test will be performed: Create a test case in UTF that creates a new user with all information provided, verify the user is created correctly

2. FRT-M1-2

   Name: Successful new user creation

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: NA

   Input: User:

- email: unitTest@test.com
- nickname: 'Mr. Unit Test'
- photoURI: http://example/com/photo.jpg
- program: 'software engineering'
- level: 1
- lectures: {'Lecture 1', 'Lecture 2'}
- event: {'Event 1', 'Event 2'}
- friends: {'unitTest1@test.com', 'unitTest2@test.com'}
- friendInvitation: {'unitTest3@test.com'}

Output: A user is successfully created with attributes above

Test Case Derivation: Verify that a new user can be created with all attributes specified

How test will be performed: Create a test case in UTF that creates a new user with all information provided, verify the user is created correctly

5.2.2 Event Module

This section will contain all unit tests for the module "Event". The tests are chosen based on common user flows and cover the most important methods of this module. The getter and setter methods are part of the unit test as well, but they will not be included in the section because they are trivial.

1. FRT-M2-1

   Name: Successful default event creation

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: NA

   Input: Event identifier, name: unit test event

   Output: An event is successfully created with following attributes:

   - name: unit test event
   - description: PLACEHOLDER

- organizer: PLACEHOLDER
- duration: DEFAULT_DURATION
- time: DEFAULT_TIME
- isPulbic: false
- location: PLACEHOLDER

Test Case Derivation: Verify that a new event can be created by just specifying the name and the rest attributes will be filled automatically

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

2. FRT-M2-2

   Name: Successful new event creation

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: NA

   Input: Event:

   - name: unit test event2
   - description: A unit test sample event
   - organizer: Team 2
   - duration: 30
   - time: 1708922625
   - isPulbic: true
   - location: Online

   Output: An event is successfully created with attributes above

   Test Case Derivation: Verify that a new event can be created with all attributes specified

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

5.2.3 Lecture Module

This section will contain all unit tests for the module "Lecture". The tests are chosen based on common user flows and cover the most important methods of this module. The getter and setter methods are part of the unit test as well, but they will not be included in the section because they are trivial.

1. FRT-M3-1

   Name: Successful default lecture creation

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: NA

   Input: Lecture identifier, code: UNIT TEST3

   Output: A lecture is successfully created with following attributes:

   - code: UNIT TEST3
   - name: PLACEHOLDER
   - instructor: PLACEHOLDER
   - time: PLACEHOLDER
   - location: PLACEHOLDER

   Test Case Derivation: Verify that a new lecture can be created by just specifying the code and the rest attributes will be filled automatically

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

2. FRT-M3-2

   Name: Successful new lecture creation

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: NA

   Input: Lecture:

   - code: UNIT TEST3
   - name: Unit test Lecture

- instructor: Team 2
- time: 12:30 - 13:30, Mon
- location: Online

Output: A lecture is successfully created with attributes above

Test Case Derivation: Verify that a new lecture can be created with all attributes specified

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

5.2.4 Pagination and Filter Module

This section will contain all unit tests for the module "Pagination and Filter". The tests are chosen based on common user flows and cover the most important methods of this module. The getter and setter methods are part of the unit test as well, but they will not be included in the section because they are trivial. The Pagination module here has a type T = Pizza, which has two string attributes: toppings and size.

1. FRT-M4-1

   Name: Successful pagination creation

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: NA

   Input: List of Pizza:

   - medium, cheese
   - large, veggie

   pageCount = 2, filterBy and filterString are null

   Output: A pagination class is created with following attributes:

   - entryList: List of pizza from input
   - filteredList: same as entryList
   - maxPage: 1

- currentPage: 1
- filterBy: null
- filterString: null

Test Case Derivation: Verify that a new pagination object can be created by just specifying a list of T, page count, filter content and filter type

How test will be performed: Create a test case in UTF that creates a new pagination with all information provided, verify the pagination is created correctly with all the attributes expected

2. FRT-M4-2

   Name: Successful pagination creation with filter

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: NA

   Input: List of Pizza:

   - medium, cheese
   - large, veggie
   - medium, pepperoni

   pageCount = 2, filterBy = 'size' and filterString = 'medium'

   Output: A pagination class is created with following attributes:

   - entryList: List of pizza from input
   - filteredList: List of medium pizza only
   - maxPage: 1
   - currentPage: 1
   - filterBy: 'size'
   - filterString: 'medium'

   Test Case Derivation: Verify that a new pagination object can be created with a filter

   How test will be performed: Create a test case in UTF that creates a new pagination with all information provided, verify the pagination with filter is created correctly with all the attributes expected

3. FRT-M4-3

   textbfName: Successful pagination addition

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: List of Pizza:

   - medium, cheese
   - large, veggie

   pageCount = 2, filterBy and filterString = null

   Input: New pizza added to the list:

   - medium, pepperoni

   Output: A pagination class is created with following attributes:

   - entryList: List of all three pizza
   - filteredList: same as entryList
   - maxPage: 2
   - currentPage: 1
   - filterBy: null
   - filterString: null

   Test Case Derivation: Verify that a new object of type T can be added to a pagination object

   How test will be performed: Create a test case in UTF that creates a new pagination with all information provided and adds a new object to the class, verify the pagination attributes are updated correctly

4. FRT-M4-4

   textbfName: Successful pagination deletion

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: List of Pizza:

   - medium, cheese
   - large, veggie

- medium, pepperoni

pageCount = 2, filterBy and filterString = null

Input: New pizza to be deleted from the list:

- medium, pepperoni

Output: A pagination class is created with following attributes:

- entryList: List of two pizza (cheese and veggie)
- filteredList: same as entryList
- maxPage: 1
- currentPage: 1
- filterBy: null
- filterString: null

Test Case Derivation: Verify that a new object of type T can be removed from a pagination object

How test will be performed: Create a test case in UTF that creates a new pagination with all information provided and deletes an object from the class, verify the pagination attributes are updated correctly

5. FRT-M4-5

   textbfName: Next page

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: List of Pizza:

   - medium, cheese
   - large, veggie
   - medium, pepperoni

   pageCount = 2, filterBy and filterString = null

   Input: Call next page method twice

   Output: Current page is updated only once:

   - currentPage: 2

Test Case Derivation: Verify that the current page increases by one when next page method is called (if the page is not last page)

How test will be performed: Create a test case in UTF that creates a new pagination with all information provided and calls next page twice verify the pagination current page is updated correctly

6. FRT-M4-6

   textbfName: Previous page

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: List of Pizza:

   - medium, cheese
   - large, veggie
   - medium, pepperoni

   pageCount = 2, filterBy and filterString = null
   Call next page method first

   Input: Call previous page method twice

   Output: Current page is updated only once:

   - currentPage: 1

   Test Case Derivation: Verify that the current page reduces by one when previous page method is called (if the page is not first page)

   How test will be performed: Create a test case in UTF that creates a new pagination with all information provided and calls previous page twice verify the pagination current page is updated correctly

7. FRT-M4-7

   textbfName: Last page

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: List of Pizza:

   - medium, cheese
   - large, veggie

- medium, pepperoni

pageCount = 2, filterBy and filterString = null

Input: Call last page method

Output: Current page is updated:

- currentPage: 2

Test Case Derivation: Verify that the current page is updated to be max page when last page method is called

How test will be performed: Create a test case in UTF that creates a new pagination with all information provided and calls last page method, verify the pagination current page is updated correctly

8. FRT-M4-8

   textbfName: First page

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: List of Pizza:

   - medium, cheese
   - large, veggie
   - medium, pepperoni

   pageCount = 2, filterBy and filterString = null
   Call next page method first

   Input: Call first page method

   Output: Current page is updated:

   - currentPage: 1

   Test Case Derivation: Verify that the current page is updated to be 1 when first page method is called

   How test will be performed: Create a test case in UTF that creates a new pagination with all information provided and calls first page method, verify the pagination current page is updated correctly

5.2.5 Authentication Module

This section will contain all unit tests for the module "Authentication". This module wraps some methods from third-party library Firebase Authentication, and these methods will be mocked in the unit tests.

1. FRT-M5-1

   Name: Get current user email

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: Current user email: 'unitTest@test.com'

   Input: NA

   Output: The email in the initial state

   Test Case Derivation: Verify that the current user is available in the system

   How test will be performed: Create a test case in UTF that creates an authentication module and calls the get current user email method with a mock of Firebase Authentication current user

2. FRT-M5-2

   Name: Is email verified

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: Current user is email verified: true

   Input: NA

   Output: The boolean value in initial state

   Test Case Derivation: Verify that the application can check email verification state

   How test will be performed: Create a test case in UTF that creates an authentication module and calls the get current email verified method with a mock of Firebase Authentication email verification value

3. FRT-M5-3

   Name: Verify email

   Type: Functional, Dynamic, Automatic, Unit

Initial State: Current user email: 'unitTest@test.com'

Input: Call verify email method

Output: An email is sent to the email address to verify the email

Test Case Derivation: Verify that the application can send verification email if requested

How test will be performed: Create a test case in UTF that creates an authentication module and calls the verify email method with a mock of Firebase Authentication email verifying method

5.2.6 Database Module

This section will contain all unit tests for the module "Database". This module wraps some methods from third-party library Firebase Database, and these methods will be mocked in the unit tests.

1. FRT-M6-1

   Name: Get value

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: NA

   Input: A valid path to a test node: '/events/public/EXPO/name'

   Output: The corresponding value of the node: EXPO

   Test Case Derivation: Verify that the module can read from the database

   How test will be performed: Create a test case in UTF that creates a database module and calls the read method with a mock of Firebase database read method

2. FRT-M6-2

   Name: Set value

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: NA

   Input: A valid path to a test node and a new value:

   - path: '/events/public/EXPO/description'

- value: ‘EXPO, yes!’

Output: The corresponding value changes to the new value

Test Case Derivation: Verify that the module can edit the database

How test will be performed: Create a test case in UTF that creates a database module and calls the set method with a mock of Firebase database set method

3. FRT-M6-3

   Name: Delete data

   Type: Functional, Dynamic, Automatic, Unit

   Initial State: NA

   Input: A valid path to a test node : ‘/events/public/EXPO’

   Output: The corresponding event is removed

   Test Case Derivation: Verify that the module can delete data from the database

   How test will be performed: Create a test case in UTF that creates a database module and calls the delete method with a mock of Firebase database delete method

5.3 Tests for Nonfunctional Requirements

This application does not have non-functional requirements to be tested with unit testing.

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-M2-1		X				
FRT-M2-2		X				
FRT-M3-1			X			
FRT-M3-2			X			
FRT-M4-1				X		
FRT-M4-2				X		
FRT-M4-3				X		
FRT-M4-4				X		
FRT-M4-5				X		
FRT-M4-6				X		
FRT-M4-7				X		
FRT-M4-8				X		
FRT-M5-1					X	
FRT-M5-2					X	
FRT-M5-4					X	
FRT-M6-1						X
FRT-M6-2						X
FRT-M6-3						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 data	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 and happens only with the user's permission	P-SC1, S-P1, COM-L1
Database Walkthrough	A code walkthrough focusing on database connection and its storage. It will be held when the database implementation is done and convince people that the database is capable of storing all user data and will clean unused data periodically	P-SE2, S-P2
Server & Data Transmission Walkthrough	A code walkthrough focusing on the server structure. It will be held when the server implementation is done and convince people that the server encrypts data before transmitting and is robust against unexpected crash	P-SC2, P-RF3
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
Code Walkthrough with Primary Reviewers	A code walkthrough focusing on Unity scripts and AR implementation. It will be held after Revision 0 is done and convince our primary reviewers that the a new target building can be added easily with our AR feature	P-SE3

Table 9: Nondynamic Test Plan Cont

Test Method	**Description**	**Related Requirement(s)**
GitHub Walkthrough with Reviewers	A walkthrough happens when the team has meetings with reviewers. During the walkthrough, the other team confirms that they can read issues the team created and also write new issues. Also, they will confirm there is a clear instruction for users	UH-L1, MS-S1, MS-S2
Document Review	The peer evaluation happens after the deadline for all important documents. During the document review, the other team points out problems and improves document quality	All
Weekly Code Review	The developer will have weekly code reviews with the team for better code quality and formatting	All
TA Review and Feedback	The developer will have meetings with the TA for every deliverable to get some feedback	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 10: **Symbolic Parameter Table**

Symbolic Parameter	**Description**	**Value**
INITIAL_USER_STATE	The default user state with all default user information	All entries empty except email, nickname and password
DEFAULT_AVATAR	The default virtual avatar	An unisex avatar with default settings
MIN_SCORE	The passing grade for a category in the survey	3.5/5
MAX_CAPACITY	The max number of simultaneous users	100
SURVEY_SAMPLE_SIZE	Sample size of the usability survey	5
SCREEN_VIEWPORTS	List of all popular mobile screen sizes	360X740, 390X844, 820X1180
RECOGNITION_TIME	Max time the system takes to recognize image	3 second
LOCATION_UPDATE_TIME	Max time the system takes for location to update	10 second
AR_ACCURACY	AR camera target recognition success rate	80%
PLACEHOLDER	Default value of attributes of type string for data structures in unit testing	NA
DEFAULT_TIME	Default unix time for events	1708922625
DEFAULT_DURATION	Default duration (in minutes) for events	30
SPECIAL_CHAR	Suspicious characters that can be used for SQL injection	| ! # $ % & / ( ) = ? @ £ § € { } . - ; ' , ⟨⟩

7.2 Usability Survey

Usability testing holds particular significance, especially for a social networking application such as this one. We intend to conduct a usability survey among a select group of students to gather feedback. Initially, we will start with a sample size of 5 to 10 participants. If we consistently receive repetitive suggestions, signalling that we've addressed the majority of issues, we will maintain this sample size. However, if new issues arise or further insights are needed, we will consider conducting another survey with a larger sample size in the future.
The survey will be composed of following three session and take about 30 minutes to complete:

7.2.1 Tasks

The participant will be completing the following tasks. They will use the project as a guest or user or administrator. Each role will have unique tasks and the role is assigned randomly at the start of a test. A task fails if the tester asks for help or spends more than 1 minute to complete a single subtask. This section is to test if the UI and layout is user-friendly and easy-to-understand.

1. Lecture
    (a) Direct from menu page to lecture page
    (b) Search for a lecture the tester asked for
    (c) Check the location of the lecture
    (d) Pin and then unpin the lecture
2. Event
    (a) Direct from map page to event page
    (b) Search for an event the tester asked for
    (c) Check the location of the event
    (d) Pin and then unpin the event
3. Activity Management

(a) Direct from menu page to event/lecture page (tester decide which page they will test)

(b) Add a new activity, where the tester will provide details

4. Map

   (a) Direct from menu page to map page

   (b) Zoom in and center the map to show the current location

5. AR Camera

   (a) Direct from menu page to AR page

   (b) Walk around and check if there exist AR object or watch the recording if the participant is not in a target building

6. User Profile

   (a) Direct from map page to user profile page

   (b) Check the program and level of the user

   (c) Edit any part of the profiles

   (d) Check the lectures and events that are pinned

7. Friend and Chat

   (a) Direct from menu page to friend page

   (b) *option 1:* Send a request to tester's account

   (c) *option 2:* Accept a request from tester

   (d) Open a chat with tester and send a message

Reminder: A participant will not do all of the tasks. They will be given a random role at the start of the test and the tasks they will have depend on the role. Administrator will perform task 3, 4, 5, 6, 7; User will perform task 1, 4, 5, 6, 7; Guest will perform task 2, 4, 6.

7.2.2 Rating

This section covers tests for some non-functional requirements. The participant will answer the following agree/disagree questions on a rating scale of 5. Here is a breakdown:

- 1: Strongly Disagree
- 2: Disagree
- 3: Neutral
- 4: Agree
- 5: Strongly Agree

1. **Immediate Visual Response when Clicking:** The application provides visual feedback when using different features or switching between pages.
2. **Appealing Colour Scheme:** The colour scheme chosen for the interface is similar to McMaster official application.
3. **No Technical or Software-specific Language:** The product does not use technical/software-specific language unless necessary.
4. **Common periods of usage:** You will be using the application mostly during school time (from 8:30 AM to 6:30 PM).
5. **Cultural Friendliness:** The application does not have any words or symbols that are offensive to you.

   If given more time, the following questions will be added when conducting a new usability survey

6. **Customer Satisfaction:** This application will be useful for my daily life and I'd like to recommend it to my friends.
7. **Loading Speed:** This application loads/updates what I need in a reasonable time.

7.2.3 Open-ended Questions

In this section we invite participants to provide their thoughts on the pros and cons of the project. They should feel free to share any feedback or insights.

- What do you think is the most difficult to use feature and why?
- Which feature will you use the most in your daily live and why?

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.