1 CMPT 371 – Team 3 Testing Document

2 Virtual Reality Medical Imaging Software with Luxsonic Technologies Inc.

3 Purpose

This document will plan what types of tests we will be running and how we will be running them each iteration. We will be using a top-down approach, as we will start with smoke tests and then work our way down through system and integration testing. Additionally, regression testing will be documented as issues arise and fixes are implemented. Coverage testing will be run near the completion of testing in each iteration, as well as user testing and acceptance testing once we are confident in the delivery of the requirements of the system. Lastly this document will contain the most up to date version of the testing matrix at all times.

4 Smoke Testing

These tests are aimed at ensuring the most basic functionality of the system. They are the first set of tests that will be run during a cloud build so that if they fail, the build fails early before it begins running more exhaustive testing.

4.1 Verify all dependencies (libraries, imports, etc)

- Check for Occulus SDK
- Check for Nsubstitute
- Check for UberLogger
- Check for DICOM folder

4.2 User gets into the main menu

Test:

- Start the system
- Test that the MainMenu member “displayGUI” is set to “true”
- Confirm that there is a visual of the main menu in the display

4.3 User gets into the workspace

Test:

- Start the system
- Generate a button press on ‘Start” in the the Main Menu
- Test that the ImageManager, Display, and Workspace classes are instantiated
- Test that the MainMenu member “displayGUI” is set to “false
- Confirm all tools and menus within the workspace exists and are available

4.4 Is the camera displaying and does it move??

Test:

- Enter the workspace
- Assert that the camera exists
- Assert that the camera moves with the Oculus

4.5 Are button clicks registering?

Test:

- Start the system
- Access all buttons within each class
- Generate a click for each one of the buttons
- Catch the click events to confirm that all button clicks are being registered.

5 System testing

The purpose of these tests is to ensure that the system is in compliance with the stakeholder's requirements.The tests are all black-box and use the same options that the user would have in the user interface.

5.1 Must-Haves

5.1.1 Initialize The Program

The user presses "Start" in the main menu and the workspace is displayed

Test 1

- Put the user in the main menu
- Click the "Start" button
- Test that the workspace has been loaded

5.1.2 Quit Application

The user presses "Exit" in the main menu and the application is closed.

Test 1

- Put the user in the main menu
- Click the "Exit" button
- Test that the application has been closed

5.1.3 Load DICOM File

The first test will test that a DICOM file loads correctly into the workspace. The test will have the user select a DICOM file to load and confirm the selection. It will then confirm that the file was loaded into the workspace.

The second test will test that an error message appears when dealing with a corrupt or non-DICOM file type. The test will have the user select a corrupt or a non-DICOM file to load to the workspace. It will then confirm that an error message is displayed to the user and that the file was not put into the workspace.

The third test will test denying the file selection. The test will have the user select files to load into the workspace and press the “Load Files” button. The user will then deny the selection on the confirmation message. It will then confirm that the user is returned to the file selection menu and that the selected files were not loaded into the workspace.

Test 1: User confirms file selection and there is a successful display in the workspace

- Put the user in the workspace
- Click the “Load Files” button
- Select DICOM files for loading
- Confirm selection by clicking “Yes”
- Test that all files selected are now loaded within the workspace

Test 2: Error Message Case

- Put the user in the workspace
- Click the “Load Files” button
- Select broken DICOM files and files that are not type DICOM
- Confirm selection by clicking “Yes”
- Test that an error message is displayed and that the broken/wrong-type of files were not attempted to be put into the workspace.

Test 3: User denies file selection and is returned to the file selection menu

- Put the user in the workspace
- Click the “Load Files” button
- Select DICOM files for displaying
- Deny selection by clicking “No”
- Test that the file selection menu is displayed and that the selected files were loaded into the workspace.

5.1.4 Exit Workspace

The first test will test that files within a workspace are returned to their default format when the workspace is exited. The test will have the user in the workspace with a few files, making format changes to a couple of them. The user will then press the “Exit Workspace” button. Confirm that the files that were changed are back to their default format.

The second test will test that workspace settings are saved when the workspace is exited. The test will have the user make a change to the settings of the workspace and then press the “Exit Workspace” button. Confirm that the settings of the workspace were saved by reopening the workspace and checking the settings values.

Test 1: Changed files are returned to default format

- Put the user in the workspace with 3 files
- Make changes to the format of 2 of the files
- Click the “Exit Workspace” button
- Test that the 2 changed files are returned to their default format and that the unchanged file is still in the default format.

Test 2: Changes to workspace settings are saved

- Put the user in the workspace
- Make changes to the workspace setting
- Click the “Exit Workspace” button
- Test that the changed workspace settings were saved by reopening the workspace and by checking all changed values.

5.1.5 Select Images to Display

The test will test that the images from the loaded DICOM files are displayed in the workspace when selected. The test will have the user select the a couple DICOM files and click the “Display Images” button. Confirm that selected images are displayed in the workspace.
Test 1

- Load multiple DICOM files into the workspace
- The user from within the workspace will select a couple DICOM files to display
- Click the “Display Images” button
- Test that the correct images are displayed in the workspace

5.1.6 Remove Images from Display

Test 1 will test that images will remove from display properly. The test will be setup by having a few images on the display in the workspace. The “Remove” button will then be toggled and the user will select images to remove from the display. Confirm that images are removed as the user selects the images.

Test 2 will test that the remove mode cannot be entered while no images are on display. The test will be setup so there are no images on display in the workspace. The user will then try to enter remove mode. Confirm that remove mode is never entered.

Test 1: Remove images from display

- Put the user in the workspace with multiple images displayed
- Toggle the “Remove” button to enter remove mode
- Test that the user is within remove mode
- Select a couple images to remove from the display
- Test that the images are removed from the display once selected
- Toggle the “Remove” button to exit remove mode
- Test that the user is no longer in remove mode.

Test 2: No images are on display

- Put the user in the workspace with no images displayed
- Try to toggle the “Remove” button to enter remove mode
- Test that remove mode was not entered, with no images on Display

5.1.7 Adjust Contrast

Test 1 tests that contrast changes are applied correctly to an image. The test will be setup with an image displayed in the workspace. Select the image, then contrast settings. Increase and decrease the contrast and confirm that the changes are applied correctly in real time.

Test 2 tests that turning off the contrast effect works. The test will be setup with an image displayed in the workspace. Select the image, then contrast settings. Increase or decrease the contrast. Toggle the contrast effect off. Confirm that the contrast settings are returned back to their default values.

Test 3 will test that the contrast settings cannot be accessed while no images are on display. The test will be setup so there are no images on display in the workspace. The user will then try to access contrast settings. Confirm that contrast settings cannot be accessed.

Test 1: User applies contrast changes to an image

- Put the user in the workspace with an image displayed
- Select an image to make contrast adjustments to
- Select contrast settings
- Test that both increasing and decreasing the contrast works correctly with changes displayed in real time.

Test 2: User does not apply contrast changes
- Put the user in the workspace with an image displayed
- Select an image to make contrast adjustments to
- Select contrast settings
- Change the contrast of the image
- Toggle the contrast effect off
- Test that the image's contrast returned to its default value.

Test 3: No images are on display
- Put the user in the workspace with no images displayed
- Try to select contrast settings
- Test that contrast setting cannot be accessed with no images on display

5.1.8 Adjust Brightness

Test 1 tests that brightness changes are applied correctly to an image. The test will be setup with an image displayed in the workspace. Select the image, then brightness settings. Increase and decrease the brightness and confirm that the changes are applied correctly in real time.

Test 2 tests that turning off the brightness effect works. The test will be setup with an image displayed in the workspace. Select the image, then brightness settings. Increase or decrease the brightness. Toggle the brightness effect off. Confirm that the brightness settings are returned back to their default values.

Test 3 will test that the brightness settings cannot be accessed while no images are on display. The test will be setup so there are no images on display in the workspace. The user will then try to access brightness settings. Confirm that brightness settings cannot be accessed.

Test 1: User applies brightness changes to an image
- Put the user in the workspace with an image displayed
- Select an image to make brightness adjustments to
- Select brightness settings
- Test that both increasing and decreasing the brightness works correctly with changes displayed in real time.

Test 2: User does not apply brightness changes

- Put the user in the workspace with an image displayed
- Select an image to make brightness adjustments to
- Select brightness settings
- Change the brightness of the image
- Toggle the brightness effect off
- Test that the image's brightness returned to its default value.

Test 3: No images are on display
- Put the user in the workspace with no images displayed
- Try to select brightness settings
- Test that brightness setting cannot be accessed with no images on display

5.1.9 Apply Filter

Test 1 tests that a filter is applied correctly to an image. The test will be setup with an image displayed in the workspace. Select the image, then filter settings. Select different filters to the image and confirm that the filters are applied correctly in real time.

Test 2 tests that turning off a filter works. The test will be setup with an image displayed in the workspace. Select the image, then filter settings. Apply a filter to an image then toggle the filter off. Confirm that the filter is no longer applied to the image.

Test 3 will test that the filter settings cannot be accessed while no images are on display. The test will be setup so there are no images on display in the workspace. The user will then try to access filter settings. Confirm that filter settings cannot be accessed.

Test 1: User applies filter changes to an image
- Put the user in the workspace with an image displayed
- Select an image to apply a filter to
- Select filter settings
- Apply different filters to the image, Testing that they are being applied correctly in real time.

Test 2: User does not apply filter changes to an image
- Put the user in the workspace with an image displayed
- Select an image to apply a filter to
- Select filter settings
- Apply a filter to an image
- Toggle the filter off
- Test that the image's filter returned to its default value.

Test 3: No images are on display

- Put the user in the workspace with no images displayed
- Try to select filter settings
- Test that filter setting cannot be accessed with no images on display

5.1.10 Zoom

Test 1 tests that zoom changes are applied correctly to an image. The test will be setup with an image displayed in the workspace. Select the image, then zoom settings. Zoom in and out of the image and confirm that the changes are applied correctly in real time.

Test 2 tests that turning off the zoom effect works. The test will be setup with an image displayed in the workspace. Select the image, then zoom settings. Zoom in or out of the picture then toggle the zoom effect off. Confirm that the zoom settings are returned back to their default values.

Test 3 will test that the zoom settings cannot be accessed while no images are on display. The test will be setup so there are no images on display in the workspace. The user will then try to access zoom settings. Confirm that zoom settings cannot be accessed.

Test 1: User applies zoom changes to an image
- Put the user in the workspace with an image displayed
- Select an image to zoom in or out of
- Select zoom settings
- Test that both zooming in and zooming out of the image works correctly with changes displayed in real time.

Test 2: User does not apply zoom changes to an image
- Put the user in the workspace with an image displayed
- Select an image to zoom in or out of
- Select zoom settings
- Zoom in or out of the image
- Toggle the zoom effect off
- Test that the images returns to its default zoom value

Test 3: No images are on display
- Put the user in the workspace with no images displayed
- Try to select zoom settings
- Test that zoom settings cannot be accessed with no images on display

5.1.11 Rotate

Test 1 tests that rotation changes are applied correctly to an image. The test will be setup with an image displayed in the workspace. Select the image, then rotation settings. Rotate the image left and right and confirm that the changes are applied correctly in real time.

Test 2 tests that turning off the rotation effect works. The test will be setup with an image displayed in the workspace. Select the image, then rotation settings. Rotate the picture left or right, then toggle the rotation effect off. Confirm that the image is returned to its default orientation.

Test 3 will test that the rotation settings cannot be accessed while no images are on display. The test will be setup so there are no images on display in the workspace. The user will then try to access rotation settings. Confirm that rotation settings cannot be accessed.

Test 1: User applies rotation changes to an image

- Put the user in the workspace with an image displayed
- Select an image to rotate
- Select rotate settings
- Test that rotating the image left and right works correctly with changes displayed in real time.

Test 2: User does not apply rotation changes to an image

- Put the user in the workspace with an image displayed
- Select an image to rotate
- Select rotate settings
- Rotate the image
- Toggle the rotate effect off
- Test that the image returns to its default rotation

Test 3: No images are on display

- Put the user in the workspace with no images displayed
- Try to select rotation settings
- Test that the rotation settings cannot be accessed with no images on display

5.1.12 Flip

Test 1 tests that flip changes are applied correctly to an image. The test will be setup with an image displayed in the workspace. Select the image, then flip settings. Flip the image backwards and upside-down and confirm that the changes are applied correctly in real time.

Test 2 tests that turning off the flip effect works. The test will be setup with an image displayed in the workspace. Select the image, then flip settings. Flip the image backwards or upside-down, then toggle the flip effect off. Confirm that the image is returned to it's default flip settings.

Test 3 will test that the flip settings cannot be accessed while no images are on display. The test will be setup so there are no images on display in the workspace. The user will then try to flip rotation settings. Confirm that flip settings cannot be accessed.

Test 1: User applies rotation changes to an image

- Put the user in the workspace with an image displayed
- Select an image to flip
- Select flip settings
- Test that flipping the image backwards or upside-down works correctly with changes displayed in real time.

Test 2: User does not apply flip changes to an image

- Put the user in the workspace with an image displayed
- Select an image to flip
- Select flip settings
- flip the image
- Toggle the flip effect off
- Test that the image returns to its default settings

Test 3: No images are on display

- Put the user in the workspace with no images displayed
- Try to select flip settings
- Test that the flip settings cannot be accessed with no images on display

5.1.13 Move Monitor

The test will test that a monitor moves correctly within the virtual environment. The test will put the user in the workspace with a monitor on display. The user will then move the monitor to different locations. Confirm that the image moves in real time to the correct locations.

Test 1: Move a monitor within the virtual environment

- Put the user in the workspace with a monitor displayed
- Select a monitor to move
- Move the monitor to several desired locations
- Test that the monitor moves to the correct locations with changes occurring in real time.

5.1.14 Resize Monitor

The test will test that a monitor resizes correctly within the virtual environment. The test will put the user in the workspace with a monitor on display. The user will then select the monitor and adjust it to different sizes. Confirm that the monitor resizes in real time to the correct size.

Test 1: Resize a monitor within the virtual environment

- Put the user in the workspace with a monitor displayed
- Select a monitor to resize
- Resize the monitor to several different sizes
- Test that the monitor adjusts to the correct size with changes occurring in real time.

5.1.15 DICOM Decoder

Test 1 will test that DICOM file decoding works. The test will have a legitimate DICOM file selected and sent to the DICOM Decoder to decipher. Confirm that the file is deciphered correctly and stored for program use.

Test 2 will test that illegitimate or corrupt files are dealt with correctly. The test will have a illegitimate or corrupt file selected and sent to the DICOM Decoder to decipher. Confirm that the DICOM Decoder sends nothing back to the program to store.

5.1.16 DICOM file folder

Test 1: Decode a legitimate DICOM file

- Select an existing legitimate DICOM file to decode
- Send the file to the DICOM Decoder
- Test that the file is deciphered correctly and that it is sent and stored correctly to the program

Test 2: Try to decode an illegitimate/corrupt DICOM file

- Select an illegitimate/corrupt DICOM file to decode
- Send the file to the DICOM Decoder
- Test that the decoder sends nothing back to the program to store

5.2 Should-Haves

5.2.1 Query the DICOM folder for an image

To test we will first check if the DICOM folder exists, then check if it contains any DICOM files, and then assert that the load screen can see a file of the same name that was loaded.

5.2.2 Display 3 images at the same time

The program should have the ability to display three images at the same time for our first incremental. To test this, the user must load at least three valid DICOM files into the workspace. Next select three of those files to display at once. Confirm that all three images are on the display with a nice visual. Confirm also that the images can be manipulated separately.

5.2.3 Anti-Motion Sickness Features

To test our added Anti-Motion Sickness features, we will need to test with multiple users. The tests should include people who are prone to motion sickness, as well as people who don't usually get motion sick. To test the features the user should first navigate around in the virtual environment without the added feature for about 10-15 minutes. They should then fill out a questionnaire about their symptoms they experienced, if any, such as eyestrain and nausea. The anti-motion sickness feature should then be added while the user takes a break until their symptoms have left. The user will then navigate around the virtual environment again with the added feature for about 10-15 minutes and then fill out a questionnaire about if the feature helped or made their symptoms worse.

5.2.4 Create 3D Models from DICOM Images to View and Manipulate

The testing of 3D Models would use many of the Must-Have tests above that are written currently for 2D. Though, they would slightly differ because they would have to test the additional dimension. To test the additional dimension, extra test cases will be added to the must-have tests. An example of one of these additional tests would be for the rotation action. There must be a test not only for just left and right rotation, but also rotation forward or backwards.

5.3 Could-Haves

Could-Haves will not have tests written for them in the original testing document. Instead, they will be appended to the document later in the iteration if the situation arises that we will be implementing one or more.

6 Integration testing

Involves testing the interactions between integrated components to ensure that their behaviours are proper. These tests a should cover the interactions of functions that call each other from the same class or separate classes via white-box testing and without the use of mocks. The UML diagram should help show some of the relationships between components.

6.1 Display calls contrast

- Create a display

Test 1: Acceptable value changes contrast
- Change contrast to another acceptable value
- Test that it has been changed

Test 2: Upper Bound changes contrast
- Change contrast to the upper bound of what is acceptable
- Test that it has been changed

Test 3: Lower bound changes contrast
- Change contrast to the lower bound of what is acceptable
- Test that it has been changed

Test 4: Greater than upper bound throws exception
- Change contrast to an unacceptable value that is greater than what is accepted
- Test that it throws an exception

Test 5: Less than lower bound throws exception
- Change contrast to an unacceptable value that is less than what is accepted
- Test that it throws an exception

6.2 Display calls rotate

- Create a display

Test1: Acceptable value changes rotate
- Change rotateto another acceptable value
- Test that it has been changed

Test2: Upper Bound changes rotate
- Change rotateto the upper bound of what is acceptable
- Test that it has been changed

Test3: Lower bound changes rotate
- Change rotateto the lower bound of what is acceptable
- Test that it has been changed

Test4: Greater than upper bound throws exception
- Change rotateto an unacceptable value that is greater than what is accepted

- Test that it throws an exception

Test5: Less than lower bound throws exception
- Change rotateto an unacceptable value that is less than what is accepted
- Test that it throws an exception

6.3 Display calls flip

- Create a display

Test1: Flip x
- Flip x
- Assert that x has been flipped

Test2: Flip y
- Flip y
- Assert that y has been flipped

6.4 Display calls zoom

- Create a display

Test 1: Acceptable value changes zoom
- Change zoom to another acceptable value
- Test that it has been changed

Test 2: Upper Bound changes zoom
- Change zoom to the upper bound of what is acceptable
- Test that it has been changed

Test 3: Lower bound changes zoom
- Change zoom to the lower bound of what is acceptable
- Test that it has been changed

Test 4: Greater than upper bound throws exception
- Change zoom to an unacceptable value that is greater than what is accepted
- Test that it throws an exception

Test 5: Less than lower bound throws exception
- Change zoom to an unacceptable value that is less than what is accepted
- Test that it throws an exception

6.5 Display calls brightness

- Create a display

Test 1: Acceptable value changes brightness
- Change brightness to another acceptable value
- Test that it has been changed

Test 2: Upper Bound changes brightness
- Change brightness to the upper bound of what is acceptable
- Test that it has been changed

Test 3: Lower bound changes brightness
- Change brightness to the lower bound of what is acceptable
- Test that it has been changed

Test 4: Greater than upper bound throws exception
- Change brightness to an unacceptable value that is greater than what is accepted
- Test that it throws an exception

Test 5: Less than lower bound throws exception
- Change brightness to an unacceptable value that is less than what is accepted
- Test that it throws an exception

6.6 Display calls resize

- Create a display

Test 1: Acceptable value changes size
- Change size to another acceptable value
- Test that it has been changed

Test 2: Upper Bound changes size
- Change size to the upper bound of what is acceptable
- Test that it has been changed

Test 3: Lower bound changes size
- Change size to the lower bound of what is acceptable
- Test that it has been changed

Test 4: Greater than upper bound throws exception

- Change size to an unacceptable value that is greater than what is accepted
- Test that it throws an exception

Test 5: Less than lower bound throws exception
- Change size to an unacceptable value that is less than what is accepted
- Test that it throws an exception

6.7 Display calls filter

Test: Test all filters
- Create a display
- Pick a filter to test
- Find what components of the image the filter changes and to what value
    - ex) filter changes the component brightness and sets it to the value: 90
- Apply the filter to the display
- Test that the values of the display match the expected value from the filter application
- Repeat test with all filters.

6.8 Display calls move

- Create a display

Test 1: Acceptable value changes location
- Change location to another acceptable value
- Test that it has been changed

Test 2: Upper Bound changes location
- Change location to the upper bound of what is acceptable
- Test that it has been changed

Test 3: Lower bound changes location
- Change location to the lower bound of what is acceptable
- Test that it has been changed

Test 4: Greater than upper bound throws exception
- Change location to an unacceptable value that is greater than what is accepted
- Test that it throws an exception

Test 5: Less than lower bound throws exception
- Change location to an unacceptable value that is less than what is accepted
- Test that it throws an exception

6.9 WorkspaceManager shows menu on display and then exits on exit

Test 1:

- Call menu
- Assert that Display is showing the menu

Test 2:

- Call exit
- Assert that Display is no longer showing menu

6.10 WorkspaceManager shows settings on display

Test 1:

- Call settings
- Assert that Display is showing the settings

6.11 WorkspaceManager adds file to ImageManager

Test 1:

- Call add file with no file selected
- Assert that no file is added

Test 2:

- Call add file with non-DICOM file
- Assert that no file is added

Test 3

- Call add file with DICOM file with non ASCII characters
- Assert that no file is added

Test 4:

- Call add file with DICOM file
- Assert that file is in DICOM list

6.12 WorkspaceManager loads file onto Display

Test 1:

- Add file
- Assert Display is empty
- Load file

- Assert Display has loaded image

6.13 WorkspaceManager removes file from ImageManager

Test1:

- Add file to Item manager
- Remove file from ItemManager
- Assert that the file no longer exists

7 Regression testing

These tests will be added to this document as bugs/errors arise. Once a bug report is made, the appropriate tests will be added here to ensure that the bug does not arise again anywhere in the code.

8 Coverage testing

SharpCover and OpenCover are to be used for statement, branch, and path coverage testing. These tools will not be included in the continuous integration as the other tests are and thus will be run once all tests are implemented in each iteration to ensure appropriate coverage and this document will be updated with those results. If coverage is not satisfactory, more tests will be appended to this document and the coverage tests will be run again.

9 User testing

User testing will require pre-test and post-test documentation to ensure the user understands the risks and requirements of the using VR software. All questions will be optional for the user to answer. These questions will be formatted in a separate document when that becomes necessary.

Pre documentation elements will include:

- The tester informing the user that they must:
    - Use hand sanitizer that is supplied
    - Accept assistance in taking the headset on and off
- How much have you used VR before?
- Are you prone to motion sickness?
- How is your current health?
- How is your eyesight?

During the testing the tester is to:

- Ask the user questions
- Take notes on their responses and comments

Post documentation will include:

- How does your head/stomach/ eyes feel?
- Were there any interactions that you did not enjoy?
- Additional comments

10 Acceptance Testing

10.1 Internal Acceptance Testing

To be done with the Project Manager once development is near completion to ensure that the system is meeting the project's requirements.

10.2 External Acceptance Testing

To be done by the Stakeholder to ensure that the system is meeting the requirements.

11 Overall Testing Matrix

	Smoke Testing	System Testing	Integration Testing	Regression Testing
Initialize the Program	X	X		
Load File		X		
Main-menu Quit System		X		
Workspace Quit System		X		
Select Images to Display		X	X	
Select Images to Remove from Display		X	X	
Adjust Contrast		X	X	
Adjust Brightness		X	X	
Apply Filter		X	X	

Zoom		X	X	
Rotate		X	X	
Flip		X	X	
Move		X	X	
Resize		X	X	
Have three images displayed at the same time		X		
Use of Oculus Rift Controllers	X	X		

More detailed testing Matrix will live at the below address during testing development to be updated after tests are implemented.The spreadsheet will be included in the deliverable at the end of each iteration.

https://docs.google.com/spreadsheets/d/1h2iEZLeFkBg_3fZ-O37UHlvriXVWkOlevt66GWLj9GI/edit#gid=0