Logo

Description automatically generated 

**Test Report**

University of Maryland Global Campus

SWEN 670 – Team A

Spring 2023

Version 1.0 

April 04, 2023

Document History

|  |  |  |
| --- | --- | --- |
| **Version** | **Issue Date** | **Changes** |
| 0.1 | 3/22/2023 | Initial Version |
| 1.0 | 4/3/2023 | Milestone 4 Submission |

[1. Executive Summary 5](#_Toc131530966)

[2. Introduction 5](#_Toc131530967)

[2.1. Purpose 5](#_Toc131530968)

[2.2. Definitions, Acronyms and Abbreviations 6](#_Toc131530969)

[2.3. Project Documents 7](#_Toc131530970)

[2.4. Requirements Traceability Matrix 7](#_Toc131530971)

[2.5. References 8](#_Toc131530972)

[2.6. Scope 8](#_Toc131530974)

[2.6.1. Out of scope 9](#_Toc131530976)

[3. Testing 9](#_Toc131530977)

[3.1. Testing Strategy 9](#_Toc131530979)

[3.2. Test Conducted 10](#_Toc131530980)

[4. Testing Steps 10](#_Toc131530981)

[4.1. Git Hub locations 10](#_Toc131530982)

[4.2. Recommended Integrated Development Environments (IDE) 11](#_Toc131530984)

[4.2.1. Android Studio 12](#_Toc131530986)

[4.2.2. VS Code 13](#_Toc131530988)

[4.3. Additional Guidance 13](#_Toc131530989)

[5. Functionality Tests Execution Findings 14](#_Toc131530990)

[5.1. Functionality Test Results Summary 14](#_Toc131530991)

[5.2. Defect Summary 15](#_Toc131530995)

[6. Functional Test Detail Procedure 16](#_Toc131530996)

[6.1. Test Case 01 – Use Case Name: View the Images as A Sphere from Hotspots 16](#_Toc131530997)

[6.2. Test Case 02 – Use Case Name: View the Images Using a VR Viewer 17](#_Toc131530999)

[6.3. Test Case 03 – Use Case Name: Move from One Transition Point to Another 18](#_Toc131531001)

[6.4. Test Case 04 – Use Case Name: The Transition from One Image to The Next Is Smooth 19](#_Toc131531002)

[6.5. Test Case 05 – Use Case Name: Zoom and Pan at each Hotspot 20](#_Toc131531004)

[6.6. Test Case 06 – Use Case Name: Sharing Link Creation 21](#_Toc131531005)

[6.7. Test Case 07 – Use Case Name: Search for Text That is Available in the Tour 22](#_Toc131531006)

[6.8. Test Case 08 – Use Case Name: Add Glow Effect to the View 23](#_Toc131531009)

[6.9. Tour Functionality 24](#_Toc131531010)

[6.9.1. Test Case 09 - Use Case Name: Create Tour 24](#_Toc131531011)

[6.10. Test Case 10 - Use Case Name: Edit Tours 25](#_Toc131531013)

[6.11. Test Case 11 - Use Case Name: Delete Tour 26](#_Toc131531014)

[7. Non-Functional Test Cases 27](#_Toc131531015)

[7.1. Test Case 12 - Use Case Name: Homepage Load Time 27](#_Toc131531016)

[7.2. Test Case 13 - Use Case Name: Image Display on Tour Page 28](#_Toc131531017)

[7.3. Test Case 14 - Successful Opening 29](#_Toc131531018)

[8. End-to-end Testing 29](#_Toc131531021)

[8.1. Usability Test: 30](#_Toc131531022)

[8.2. Reliability Test: 30](#_Toc131531023)

[8.3. Recovery Testing: 30](#_Toc131531024)

[9. Systems Tested 31](#_Toc131531026)

[10. Test Summary Report 31](#_Toc131531027)

[11. Conclusion 31](#_Toc131531028)

# Executive Summary

The purpose of this EXSUM is to allow senior executives and future testing teams an accurate summary of the overall assessment of ViroTour’s latest release.

The tests found in this report were conducted in Android Studio, and VS Code showing the code base works well with multiple IDEs. The Test were conducted by the Test Team which consists of Nicholas Platt, Jude Ibe, Shawn Kagwa, and Jacob Lynn. The testing began simultaneously with the application development starting on MARCH 15, 2023, and continued after the final code public release was finished on MARCH 30, 2023.  
  
 The ViroTour application had a successful test rate of 64%. The testing suite consists of 14 total test cases, of which 9 passed. The main passing features include UI elements, tour navigation, and traversing between images using transitional hotspots. Of the remaining 36% of tests which failed, 80% were a result of the back-end API not being ready, and the remaining 20% of failures being a result of a feature being only partially completed.

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE NUMBER** | **TEST REPORT PARAGRAPH** | **TEST NAME** | **RESULT** |
| 06 | 6.6 | Share View | FAIL |
| 07 | 6.7 | Search by Text | FAIL |
| 09 | 6.9 | Create Tour | FAIL |
| 10 | 6.10 | Edit Tour | FAIL |
| 11 | 6.11 | Delete Tour | FAIL |

*Table 1 – Functionality Test Results Failures*

Our assessment is that the application has stable performance, intuitive and easy-to-use features for tour creation and navigation, accurate and simple search functionality, and when the API is completed, successful customization of the application. We recommend the use of Android Studio or Visual Studio Code for testing the application. We express that it is crucial that the problems with the API and back-end features are addressed, however once complete the application is ready for further development, testing, and market deployment.

# Introduction

# Purpose

The purpose of this test report is to provide a summary of tests conducted and results of the ViroTour application's functionality, usability, and reliability.

The application is a standalone product that enables users to create virtual tours by uploading pictures and utilizing built-in algorithms. The report will cover various aspects of the application, including navigation, customization, and search functionality. The testing is focused on manual testing with automated testing reserved for potential follow-on projects when the application grows to the point of making this a necessity. The overall goal being to reduce risk and improve the quality of the software.

# Definitions, Acronyms and Abbreviations

The following are definitions, acronyms and abbreviations that might be mentioned in this document and need more clarity.

* Azure - Microsoft Azure, often referred to as Azure, is a cloud computing platform operated by Microsoft that provides access, management, and development of applications and services via around the world-distributed data centers.
* IDE – Integrated Development Environment
* iOS – iPhone Operating System
* Flutter - is an open source framework by Google for building beautiful, natively compiled, multi-platform applications from a single codebase.
* Hotspots - Images with their associated transition to new images.
* STP – Software Test Plan
* Team B – Refers to concurrent and collaborative team working on the back-end elements of ViroTour.
* UMGC – University of Maryland Global Campus
* VR – Virtual Reality
* VS Code: Visual Studio Code

# Project Documents

There are various documents created for this effort to provide the stakeholders, namely the project team, the client, and external users with sufficient information and understanding for the success of the project. These documents are summarized below.

**The following documents are included in the entire documentation package:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Document** | **Version** | **Date** |
| 1 | Project Management Plan (PMP) | 4.0 | 4/4/2023 |
| 2 | Software Requirements Specification (SRS) | 4.0 | 4/4/2023 |
| 3 | Technical Design Document (TDD) | 3.0 | 4/4/2023 |
| 4 | Software Test Plan (STP) | 2.0 | 4/4/2023 |
| 5 | Programmers Guide (PG) | 2.0 | 4/4/2023 |
| 6 | Deployment and Operations (DevOps/Runbook) | 2.0 | 4/4/2023 |
| 7 | User Guide (UG) | 1.0 | 4/4/2023 |
| 8 | Test Report (TR) | 1.0 | 4/4/2023 |

*Table 2.3 - Project Documents*

# Requirements Traceability Matrix

The purpose of this requirements traceability matrix is to provide a mapping between the requirements of the system under test and the tests that were performed to verify those requirements. It helps to ensure that all requirements have been tested, and that all tests can be traced back to specific requirements.

|  |  |  |
| --- | --- | --- |
| Requirement # | Requirement Description | Test Case # |
| REQ A 1.1 | Allow the user to view images as a panoramic view from a hotspot. | 01  (Section 6.2) |
| REQ A 1.2 | View panoramic images using a virtual reality (VR) view. | 02  (Section 6.2) |
| REQ A 1.3 | Allow user the ability to transition between images within a tour. | 03  (Section 6.3) |
| REQ A 1.4 | Image transition is smooth. | 04  (Section 6.4) |
| REQ A 1.5 | Zoom and pan ability with each image. | 05  (Section 6.5) |
| REQ A 2.1 | Ability to share view by long pressing on an image which will create a shareable link to that location. | 06  (Section 6.6) |
| REQ A 2.2 | Search text that is available in the tour. | 07  (Section 6.7) |
| REQ A 2.3 | Add a glow effect to an image which changes the brightness to more easily view the image. | 08  (Section 6.8) |
| REQ A 3.1 | Create a tour by supplying name, description, and images that are compiled into a tour. | 09  (Section 6.9) |
| REQ A 3.1 | Edit an existing tour which gives the user the ability to update the name and description of the tour. | 10  (Section 6.10) |

*Table 2.4 - Requirements Traceability Matrix*

# References

Assadullah, M. (2023). Software Engineering Project. Retrieved from: [https://learn.umgc.edu/d2l/home/732302](https://learn.umgc.edu/d2l/home/732302 )

# Scope

The purpose of this test report is to document the testing efforts and results for the ViroTour application. The scope of this test report includes the following:

1. Functional testing of all major features of the ViroTour application.
2. Testing of user interface (UI) elements and navigation.
3. Verification of compatibility with major web browsers and mobile devices.
4. Verification of system performance under normal and peak load conditions.
5. Security testing to ensure data privacy and protection.
6. Verification of the accuracy and completeness of the ViroTour API.
7. Testing of data validation and error handling.
8. Verification of compliance with applicable accessibility standards.
9. Verification of compliance with applicable industry standards and regulations.
10. Testing of any custom or third-party integrations.

The test report will provide a comprehensive overview of the testing process and the results obtained, as well as any issues or defects identified during testing. The report will only include the features developed and any features that were not completed will be documented in the SRS and TDD Reports. The report will be used to evaluate the readiness of the ViroTour application for release and to guide any necessary improvements or modifications to the application.

# Out of scope

Out of Scope for Team A include unit testing, penetration testing, performance testing. Additionally, to ensure success data has been staged to simulate a data feed from a server on the back end. Aspects dependent on the back end will be tested by Team B.

# Testing

# Testing Strategy

The testing and verification of the results will be part of the developers' work activity during development of the features in the ViroTour Application. The application will be formally tested when it has been determined that the application functionality is ready to be tested.

The application will have manual tests that can be proven to work when the output and input are known. All the tests must be completed in Chrome, Android, and iOS, but where it is applicable, some features work best in Android. Manual testing will be completed after features have been deemed coding complete and any defects will be recorded in this document, as well as documented on GitHub for review by the development team. These defects will be categorized and fixed in order of priority to be decided by the Product Owner.

The following versions were used throughout testing:

* Google Chrome – Version 111.0.5563
* Android OS – Version 13
* Apple iOS – Version 16.3

The status of the tests will individually be assessed as a “**PASS” or a “FAIL”.** Any failures will also have an “Additional Information” row that includes details about how/why a test is failing.

# Test Conducted

The test suite for the ViroTour application consists of 14 total test cases that is described in the Test Results and Summary section. A list of tests and their associated description in the ViroTour Software Test Plan (STP) is available in the Appendix. To ensure the developers are not testing and signing off on their own feature, the Test Manager will sign off on any test that is not a group test. A group test is performed through a video call where all application team members can view the software functioning as intended.

# Testing Steps

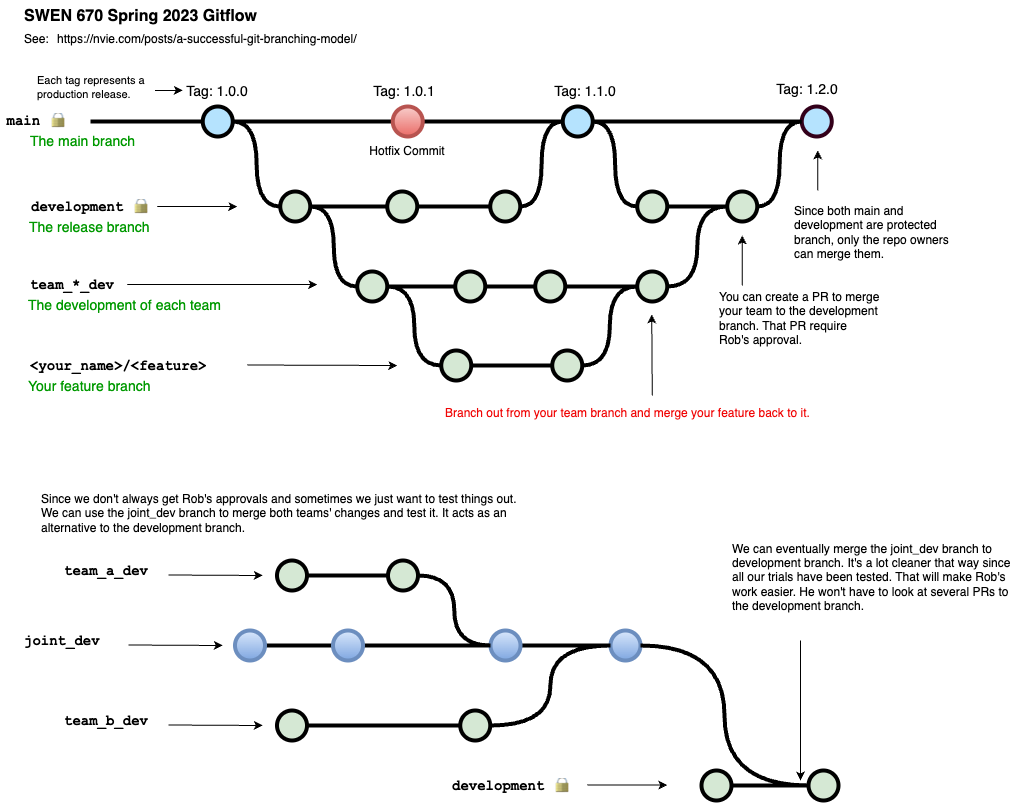
# Git Hub locations

We have made every effort to have working code merged to the main branch, however some features have not successfully been merged. For the features that only work on their own branch, the below diagram can be used for testing as well as information that can be used to merge the branches in the future.

|  |  |  |
| --- | --- | --- |
| **Branch** | **Features Available to Test\*** | **Lead Developer on Branch** |
| main | List Tours, Search, Create Tour, Edit Tour, Hamburger Menu, Wheel Menu, Tour Page |  |
| glow\_feature\_backend | Glow feature -> development | Tilahun Abreha |
| view\_tour\_vr | VR view -> development | Fedor Menchukov |

*Table 4.1 – GitHub Locations*

\*: After the projet ends, only main and development are kept, the rest will be deleted.

*Figure 3 – Gitflow Diagram*

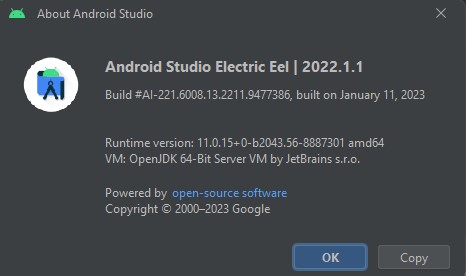
# Recommended Integrated Development Environments (IDE)

The ViroTour application is built with functionality that will provide a central interface which aids in application testing. The Integrated Development Environment will consist of the source code editor, a build automation tool, and a debugger. The importance of an IDE in the ViroTour application testing process cannot be overstated, as the application has not been published, testers must execute the application’s code to run the application. The IDE streamlines the workflow and enhances the productivity of developers. IDEs offer a comprehensive set of tools inside one unified platform, which reduces development time and improves software quality.

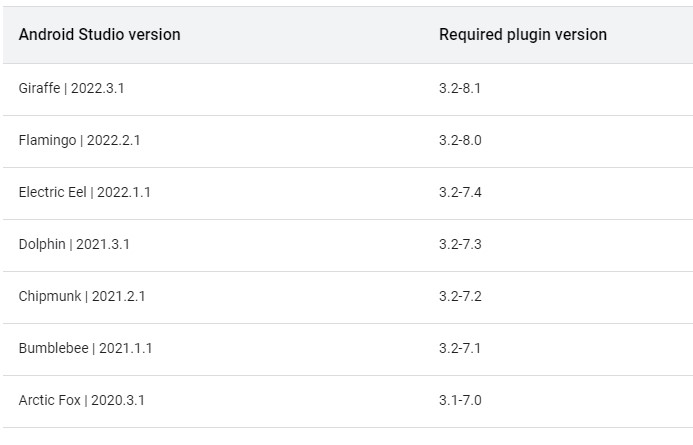
# Android Studio

Android Studio provides a unified environment where applications can be built for Android applications. The structured code allows division of projects into units of functionality that can be built independently, tested, and debugged. Considering that Flutter is a versatile software framework designed for creating high-quality, high-performance mobile and desktop applications, which cater to a wide range of operating systems, it is crucial to utilize an IDE that fully supports its capabilities. Android Studio is an excellent choice for developing Flutter applications for the Android platform.

When using Android Studio tests were conducted in the latest stable release, Electric Eel.



*Figure 4.2.1a: Electric Eel*

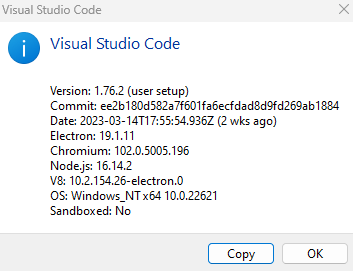


*Figure 4.2b: Versions of Android Studio*

# VS Code

Visual Studio Code, also commonly referred to as VS Code, is a source-code editor made by Microsoft with the Electron Framework, for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.

Tests conducted in VS Code were using Version: 1.76.2. And installed extensions include Dart, Flutter. Interactions with Git can be conducted through the VS Code Terminal.



*Figure 4.2.2: VS Code Version*

# Additional Guidance

More information including troubleshooting steps can be found on the accompanying Programmer’s Guide.

# Functionality Tests Execution Findings

# Functionality Test Results Summary

Below are the tests that were carried out showing the respective results of the different test cases.

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE NUMBER** | **TEST REPORT PARAGRAPH** | **TEST NAME** | **RESULT** |
| 01 | 6.1 | Viewing the image as a sphere from hotspots | PASS |
| 02 | 6.2 | VR View Functionality | PASS |
| 03 | 6.3 | Transition Between Images | PASS |
| 04 | 6.4 | Smooth Transition Between Images | PASS |
| 05 | 6.5 | Image Zoom | PASS |
| 06 | 6.6 | Share View | FAIL |
| 07 | 6.7 | Search by Text | FAIL |
| 08 | 6.8 | Image Glow Effect | PASS |
| 09 | 6.9 | Create Tour | FAIL |
| 10 | 6.10 | Edit Tour | FAIL |
| 11 | 6.11 | Delete Tour | FAIL |
| 12 | 6.12 | Homepage Load Time | PASS |
| 13 | 6.13 | Image Display on Home page | PASS |
| 14 | 6.14 | Successful Opening | PASS |

*Table 5.1 – Functionality Test Results Summary*

# Defect Summary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Date First Detected** | **Defect ID** | **Description** | **Severity** | **Priority** | **Status** |
| March 20, 2023 | 01 | Test Case 06 – Share View | High | Medium | This feature was not fully completed. The button was added to the wheel menu, but no functionality was developed for this button. |
| March 20, 2023 | 02 | Test Case 07 - Search by Text | Low | Medium | The feature is fully implemented and has been tested to work with hardcoded API responses. The feature does not currently pass when attempting to work with the API. |
| March 18, 2023 | 03 | Test Case 09 – Create Tour | Low | High | This feature is fully implemented in the front-end code. The API currently does not act as expected and leads to the feature not working as expected. |
| March 18, 2023 | 04 | Test Case 10 – Edit Tour | Low | High | This feature is fully implemented in the front-end code. The API currently does not act as expected and leads to the feature not working as expected. |
| March 18, 2023 | 05 | Test Case 11 – Delete Tour | Low | High | This feature is fully implemented in the front-end code. The API currently does not act as expected and leads to the feature not working as expected. |

# Functional Test Detail Procedure

# Test Case 01 – Use Case Name: View the Images as A Sphere from Hotspots

|  |  |
| --- | --- |
| **Description** | This use case allows users to view images as a sphere from hotspots. |
| **Requirements** | The tour must have images with location hotspots. |
| **Prerequisites** | The user must have access to ViroTour. |
| **Test Steps** | 1. Click on a hotspot. 2. Try to view the image without clicking on a hotspot. 3. Move the cursor within the image. 4. Zoom in and out using the mouse wheel or pinch gesture, applicable to mobile. |
| **Expected Output** | 1. The image should be displayed in a sphere. 2. The view should change accordingly, allowing the user to explore. 3. The image should enlarge or reduce in size, maintaining the sphere. |
| **Assumptions** | 1. Location hotspots are accurately placed in the virtual tour. 2. There are no issues with ViroTour. 3. The user has a mouse or a touchscreen device. |
| **Test #1** |  |
| **Date:** | **MARCH 20, 2023** |
| **Name of Tester:** | **Jacob Lynn** |
| **Signed off by:** | **Nick Platt** |
| **Result:** | **PASS** |
| **Test #2** |  |
| **Date:** | **MARCH 24, 2023** |
| **Name of Tester:** | **Shawn Kagwa** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **PASS** |
| **Test #3** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **PASS** |

# Test Case 02 – Use Case Name: View the Images Using a VR Viewer

|  |  |
| --- | --- |
| **Description** | Verify that users can view the images using a VR viewer. The user will be able to enjoy a virtual reality experience of tours. |
| **Requirements** | A tour must be available. |
| **Prerequisites** | 1. A virtual tour has been created with selected images. 2. The user owns a VR headset. |
| **Test Steps** | 1. User selects the VR View button. 2. The tour is started in VR View 3. Observe if the image is displayed in the VR viewer |
| **Expected Output** | 1. The current location’s image is displayed in VR View. |
| **Assumptions** | A VR Viewer headset is connected. |
| **Test #1** |  |
| **Date:** | **MARCH 20, 2023** |
| **Name of Tester:** | **Jacob Lynn** |
| **Signed off by:** | **Nick Platt** |
| **Result:** | **PASS** |
| **Test #2** |  |
| **Date:** | **MARCH 24, 2023** |
| **Name of Tester:** | **Shawn Kagwa** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **PASS** |
| **Test #3** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **PASS** |

# Test Case 03 – Use Case Name: Move from One Transition Point to Another

|  |  |
| --- | --- |
| **Description** | Verify that users can move from one hotspot to another. |
| **Requirements** | User selects a hotspot or begins automatic tour. |
| **Prerequisites** | The user selects a hotspot or begins an automatic tour. |
| **Test Steps** | 1. The user selects a hotspot to move to. 2. The tour moves to the next location’s image. |
| **Expected Output** | The user can view tour hotspots or images one after another by clicking on the transition points. When a user clicks/taps on a transition point, he should be moved to another location or image. |
| **Assumptions** | The user selects the images/hotspots manually. |
| **Test #1** |  |
| **Date:** | **MARCH 20, 2023** |
| **Name of Tester:** | **Jacob Lynn** |
| **Signed off by:** | **Nick Platt** |
| **Result:** | **PASS** |
| **Test #2** |  |
| **Date:** | **MARCH 24, 2023** |
| **Name of Tester:** | **Shawn Kagwa** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **PASS** |
| **Test #3** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **PASS** |

# Test Case 04 – Use Case Name: The Transition from One Image to The Next Is Smooth

|  |  |
| --- | --- |
| **Description** | Verify that the transition from one image to the next image is smooth, transition time should be less than a second and transitioning images should have a blur effect applied on them. |
| **Requirements** | 1. Transitioning from one image to the next image takes the system no longer than a second. 2. Transitioning between images is smoothed using the blurring effect. |
| **Prerequisites** | 1. The viewed tour has multiple images. 2. The images have transitional hotspots |
| **Test Steps** | 1. Open the app and navigate to any image in the tour. 2. Measure the time it takes to transition from one image to the next. 3. Repeat step 3 multiple times as you record the time each transition takes. 4. Check if the blur effect is present during the transitions. |
| **Expected Output** | The average time each transition takes is less than one second.  The blur effect is present in between transitions. |
| **Assumptions** | The transition time should be constant even when there are many images in a tour. |
| **Test #1** |  |
| **Date:** | **MARCH 20, 2023** |
| **Name of Tester:** | **Jacob Lynn** |
| **Signed off by:** | **Nick Platt** |
| **Result:** | **PASS** |
| **Test #2** |  |
| **Date:** | **MARCH 24, 2023** |
| **Name of Tester:** | **Shawn Kagwa** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **PASS** |
| **Test #3** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **PASS** |

# Test Case 05 – Use Case Name: Zoom and Pan at each Hotspot

|  |  |
| --- | --- |
| **Description** | Verify that users can zoom and pan at each hotspot. |
| **Requirements** | Ability to zoom in and out of an image. |
| **Prerequisites** | The user is viewing an image. |
| **Test Steps** | 1. User opens the ViroTour Application and begins a tour. 2. The user zooms in and out on an image using a mouse action, pinch gesture, or the onscreen zoom bar. |
| **Expected Output** | The user can zoom in and out of a picture, enlarging it or making it smaller. |
| **Assumptions** | The zoom functionality works on all respective platforms; mobile, desktop, and web applications.  Zoom feature controls behave in a similar manner across all applications. |
| **Test #1** |  |
| **Date:** | **MARCH 20, 2023** |
| **Name of Tester:** | **Jacob Lynn** |
| **Signed off by:** | **Nick Platt** |
| **Result:** | **PASS** |
| **Test #2** |  |
| **Date:** | **MARCH 24, 2023** |
| **Name of Tester:** | **Shawn Kagwa** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **PASS** |
| **Test #3** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **PASS** |

# Test Case 06 – Use Case Name: Sharing Link Creation

|  |  |
| --- | --- |
| **Description** | Verify that users can share a location inside a tour. |
| **Requirements** | The user is able to share a link to the currently viewed image. |
| **Prerequisites** | 1. The user is viewing an image inside a tour. |
| **Test Steps** | 1. The user selects or opens an image. 2. The user clicks on the wheel menu, then selects the “Share” button. 3. A link is generated and copied by the user. 4. The copied link opens the tour and navigates directly to the shared image location. |
| **Expected Output** | The application will generate a unique sharable location that will allow other users to access the same shared location on other devices. |
| **Assumptions** | The API has an endpoint that accepts the tour id and currently viewed image data to create a link. |
| **Test #1** |  |
| **Date:** | **MARCH 20, 2023** |
| **Name of Tester:** | **Jacob Lynn** |
| **Signed off by:** | **Nick Platt** |
| **Result:** | **FAIL** |
| **Test #2** |  |
| **Date:** | **MARCH 24, 2023** |
| **Name of Tester:** | **Shawn Kagwa** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **FAIL** |
| **Test #3** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **FAIL** |
| **Additional Information** | **This feature was not fully completed. The button was added to the wheel menu, but no functionality was developed for this button.** |

# Test Case 07 – Use Case Name: Search for Text That is Available in the Tour

|  |  |
| --- | --- |
| **Description** | Verify that the users can search for text that has been extracted from the tour’s uploaded images. |
| **Requirements** | The search string of text must be accurately matched with the text displayed in an image’s informational hotspot. |
| **Prerequisites** | All images have been uploaded and processed. |
| **Test Steps** | 1. The user types a search term on the search bar. 2. The user selects a single item/object in returned results to view the location of the text. |
| **Expected Output** | The system should return a list of objects if the search term is present in any of the objects.  Tapping one of the objects should take you to the location of the text. |
| **Assumptions** | 1. The application should have the ability to correct errors that may be present in the search term. An error in the search term should still yield the intended search outcomes which would include spelling errors and other grammar errors. 2. The application should extract text from images on upload and index them in the database. |
| **Test #1** |  |
| **Date:** | **MARCH 20, 2023** |
| **Name of Tester:** | **Jacob Lynn** |
| **Signed off by:** | **Nick Platt** |
| **Result:** | **FAIL** |
| **Test #2** |  |
| **Date:** | **MARCH 24, 2023** |
| **Name of Tester:** | **Shawn Kagwa** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **FAIL** |
| **Test #3** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **FAIL** |
| **Additional Information** | **The feature is fully implemented and has been tested to work with hardcoded API responses. The feature does not currently pass when attempting to work with the API.** |

# Test Case 08 – Use Case Name: Add Glow Effect to the View

|  |  |
| --- | --- |
| **Description** | Verify that users can add a glow effect to the view. |
| **Requirements** | The user can customize the tour lighting effect by activating the glow effect functionality. |
| **Prerequisites** | The tour is opened and positioned at the first hotspot. |
| **Test Steps** | 1. Open a tour and be positioned at the first hotspot. 2. Click on the wheel menu and select glow effect. 3. Move the slider on the bar from left to right to apply varying degrees of brightness. |
| **Expected Output** | The brightness level on a tour can be adjusted to the specified intensity. |
| **Assumptions** | A tour has been opened and is positioned at the first image/hotspot. |
| **Test #1** |  |
| **Date:** | **MARCH 18, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **PASS** |
| **Test #2** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **PASS** |
| **Test #3** |  |
| **Date:** | **MARCH 29, 2023** |
| **Name of Tester:** | **Nick Platt** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **PASS** |

# Tour Functionality

# Test Case 09 - Use Case Name: Create Tour

|  |  |
| --- | --- |
| **Description** | Verify that users can create a tour |
| **Requirements** | User logs into the application and navigates to the Hamburger Menu. |
| **Prerequisites** | The Application is open |
| **Test Steps** | 1. Click on the hamburger menu to open a dropdown list of menu options. 2. Select the create tour option to open the create tour screen. 3. Fill in the name and the description in the input fields. 4. Upload tour images using the select images button. 5. Click save to upload to create a tour or cancel to cancel the create tour process. |
| **Expected Output** | 1. A new tour is created. 2. The tour is visible in the tours list view.. |
| **Assumptions** | 1. The database is correctly implemented, and data can be saved and retrieved. 2. Anyone who has access to the application can refresh to view the newly added tour. |
| **Test #1** |  |
| **Date:** | **MARCH 18, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** |  |
| **Result:** | **FAIL** |
| **Test #2** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** |  |
| **Result:** | **FAIL** |
| **Test #3** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Nick Platt** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **FAIL** |
| **Additional Information** | **This feature is fully implemented in the front-end code. The API currently does not act as expected and leads to the feature not working as expected.** |

# Test Case 10 - Use Case Name: Edit Tours

|  |  |
| --- | --- |
| **Description** | Verify that a user can edit textual information related to a tour. |
| **Requirements** | User logs into the application and navigates to the Hamburger Menu. |
| **Prerequisites** | 1. There is at least one available tour. |
| **Test Steps** | 1. The process of editing a tour consists of changing associated descriptive information (name and description) of a tour. Open the ViroTour Application. This should open a view with a list of all tours.   Open a tour you wish to edit..   1. Click edit.   Make the desired changes to the name and/or description.   1. Click save to update the information. |
| **Expected Output** | The changes made to the tours are visible to people viewing the tour. |
| **Assumptions** | 1. The database is correctly implemented, and data can be saved and retrieved. 2. Anyone who has access to the application can refresh to view the changes made |
| **Test #1** |  |
| **Date:** | **MARCH 18, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** |  |
| **Result:** | **FAIL** |
| **Test #2** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** |  |
| **Result:** | **FAIL** |
| **Test #3** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Nick Platt** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **FAIL** |
| **Additional Information** | **This feature is fully implemented in the front-end code. The API currently does not act as expected and leads to the feature not working as expected.** |

# Test Case 11 - Use Case Name: Delete Tour

|  |  |
| --- | --- |
| **Description** | Verify that a user can delete a tour |
| **Requirements** | User logs into the application and navigates to the Hamburger Menu |
| **Prerequisites** | 1. There is at least one available tour. |
| **Test Steps** | 1. Navigate to the tour list page   Select a tour by clicking the edit icon on the right of the tour   1. Click the red “Delete” button. A pop-up will be shown for the user to verify that they would like to delete the tour. 2. Select “Delete” in the pop-up. 3. A message will display any necessary information that was received from the server upon deletion of the tour. |
| **Expected Output** | The tour is removed from the tour list. |
| **Assumptions** | 1. The database is correctly implemented, and data can be deleted. 2. Anyone who has access to the application can refresh to view the changes made. |
| **Test #1** |  |
| **Date:** | **MARCH 18, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** |  |
| **Result:** | **FAIL** |
| **Test #2** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** |  |
| **Result:** | **FAIL** |
| **Test #3** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Nick Platt** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **FAIL** |
| **Additional Information** | **This feature is fully implemented in the front-end code. The API currently does not act as expected and leads to the feature not working as expected.** |

# Non-Functional Test Cases

# Test Case 12 - Use Case Name: Homepage Load Time

|  |  |
| --- | --- |
| **Description** | Verify the homepage load time of the application |
| **Requirements** | The ViroTour application homepage being able to load within 5 seconds. |
| **Prerequisites** | The user opens the ViroTour application in any web browser or mobile device. |
| **Test Steps** | 1. The user opens a web browser and inputs the URL of ViroTour or opens the ViroTour application on a mobile device. 2. The ViroTour homepage renders completely. |
| **Expected Output** | The ViroTour homepage should render completely within 5 seconds. |
| **Assumptions** | The user must be using a Chromium-based browser (Google Chrome/Microsoft Edge), or a mobile device (iOS/Android). |
| **Test #1** |  |
| **Date:** | **18MAR2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** | **Nick Platt** |
| **Result:** | **PASS** |
| **Test #2** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** | **Nick Platt** |
| **Result:** | **PASS** |
| **Test #3** |  |
| **Date:** | **MARCH 29, 2023** |
| **Name of Tester:** | **Jacob Lynn** |
| **Signed off by:** | **Nick Platt** |
| **Result:** | **PASS** |

# Test Case 13 - Use Case Name: Image Display on Tour Page

|  |  |
| --- | --- |
| **Description** | Verify the image is displayed for each location in a tour. |
| **Requirements** | The user should see the images displayed as a sphere at each location. |
| **Prerequisites** | The user opens ViroTour app on their device and selects a tour. |
| **Test Steps** | 1. The user opens the app. 2. The user selects a tour. |
| **Expected Output** | The user sees the images displayed as a sphere at each location of the tour. |
| **Assumptions** | The user must be using a Chromium-based browser (Google Chrome/Microsoft Edge), or a mobile device (iOS/Android). |
| **Test #1** |  |
| **Date:** | **MARCH 18, 2023** |
| **Name of Tester:** | **Jacob Lynn** |
| **Signed off by:** | **Nick Platt** |
| **Result:** | **PASS** |
| **Test #2** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** | **Nick Platt** |
| **Result:** | **PASS** |
| **Test #3** |  |
| **Date:** | **MARCH 29, 2023** |
| **Name of Tester:** | **Nick Platt** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **PASS** |

# Test Case 14 - Successful Opening

|  |  |
| --- | --- |
| **Description** | Verify the ViroTour application opens successfully |
| **Requirements** | User selects one of the virtual tours from View Tours menu |
| **Prerequisites** | The successful loading of ViroTour home page. |
| **Test Steps** | 1. The user selects View Tours from the Hamburger Menu 2. The ViroTour application displays the list of tours. 3. The user selects the desired tour. 4. The ViroTour application displays the selected tour |
| **Expected Output** | The ViroTour application displays a tour matching the selection from the menu |
| **Assumptions** | 1. The home page is loaded to allow the user to display screen items. |
| **Test #1** |  |
| **Date:** | **MARCH 18, 2023** |
| **Name of Tester:** | **Jacob Lynn** |
| **Signed off by:** | **Nick Platt** |
| **Result:** | **PASS** |
| **Test #2** |  |
| **Date:** | **MARCH 25, 2023** |
| **Name of Tester:** | **Group Test** |
| **Signed off by:** | **Nick Platt** |
| **Result:** | **PASS** |
| **Test #3** |  |
| **Date:** | **MARCH 29, 2023** |
| **Name of Tester:** | **Nick Platt** |
| **Signed off by:** | **Jacob Lynn** |
| **Result:** | **PASS** |

# End-to-end Testing

End-to-end testing of the entire application or system, including all its components and subsystems, to ensure that it works as expected from start to finish. During this testing, Android Studio was used simulating real-world physical devices, potential scenarios, and user interactions to validate the system's functionality, performance, and usability.

# Usability Test:

* User Interface: The application was tested to ensure that the user interface is easy to use and understand.
* Tour Navigation: The application was tested to ensure that the tour navigation is intuitive and easy to understand.
* Search Functionality: The application was tested to ensure that the search functionality is accurate and easy to use.

# Reliability Test:

* Stability: The application was tested to ensure that it is stable and does not crash during use by having multiple users request tours at the same time. The application continued to work without a performance drop due to the scalability of Azure.
* Editing: The application was tested to ensure that tour names and descriptions can be edited successfully.

# Recovery Testing:

* Identified potential failure scenarios such as power loss, network connectivity loss, memory allocation errors, etc.
* Simulated each failure scenario and observed how the application behaves.
* Evaluated the application's ability to recover from the failure and continue to function as intended.

Conclusion: The End-to-End testing performed on the ViroTour application demonstrates that it has a robust recovery mechanism that allows it to recover from various failure scenarios and continue to function as intended. The testing results confirm that the application can handle unexpected errors and failures without compromising its functionality. Based on these results, we conclude that the ViroTour application is reliable and resilient.

# Systems Tested

|  |  |
| --- | --- |
| **System** | **Result** |
| Web - Chrome | All functions worked as designed. Extensively tested by group, Jacob, Nick, and Shawn. |
| Android | All functions worked as designed. Extensively tested by group, Jacob, Shawn, and Nick |
| iOS | All functions worked as designed however the speed of transitions and movements are less fluid on iOS in comparison to Chrome and Android. These features were tested by Viet Nguyen. |

# Test Summary Report

The ViroTour application underwent testing to assess its functionality, usability, and reliability. The testing was conducted on Web, Android, and iOS platforms on multiple devices, and the application performed well across all platforms. The test suite for the ViroTour application consisted of 13 total test cases, covering different aspects of the application, including navigation, customization, and search functionality, of which 9 passed. The main passing features include UI elements, tour navigation, and traversing between images using transitional hotspots. Of the remaining 36% of tests which failed, 80% were a result of the back-end API not being ready, and the remaining 20% of failures being a result of a feature being only partially completed.

The results of the testing showed that the ViroTour application was stable and performed well. The application's tour creation and navigation were intuitive and easy to use, and the search functionality was accurate and easy to use. The customization features of the application were also tested and when the API is finished, it should be tested again.

The Test Team recommends the use of two Integrated Development Environments (IDE), Android Studio or VS Code, to aid in the testing process. The IDEs streamlines the workflow and enhances the productivity of developers, reducing development time and improving software quality.

# Conclusion

The ViroTour application performed as expected during testing. Some features did fail (Location Sharing, Create Tour, Edit Tour, and Delete Tour) due to issues with the ViroTour API. The Create Tour, Edit Tour, and Delete Tour features were fully developed by the front-end developers, but issues with the API led to failures. The Location Sharing feature was not fully developed and additional work is required.

Overall, the ViroTour application is a well-designed and well-executed product that is near meeting the needs of its intended users. The results of the testing provide confidence that the application is nearly ready for release, and any defects discovered during testing were categorized and documented in order of priority. Additional work to fully implement the front-end features with the back-end API being the main work that needs to be completed. The recommendations for IDEs and the Programmer's Guide further support the development of the application, providing a comprehensive set of tools for the developers.