UNITED

Michigan State University

Team United Airlines Safety

Virtual Reality Aircraft Walkaround

Project Plan

Spring 2020

United Airlines Sponsors

Amadou Anne

Craig Bennett

Rick Brown

Shiju Samuel

Mike Wisniewski

Michigan State Capstone Team

Caitlin Brown

Jiachen Lin

Ellie Locatis

Jacob Turcano

Cheney Wang

Table of Contents

Executive Summary

United Airlines is a well-known commercial airline headquartered in Chicago, Illinois. With a fleet of 1,300 aircrafts, the company operates around 4,600 flights each day from over 350 airports. These flights service over a hundred million passengers annually. The safety of these millions of passengers is a top priority for United Airlines. To this end, thousands of technicians are employed to inspect and maintain the aircraft. One of their key inspection tasks is the aircraft walkaround, which is performed to identify possible issues with a plane prior to aircraft operation.

To help ensure the safety and successful operation of its flights, United Airlines technicians perform a walkaround before every flight. In a walkaround, a plane is inspected for any hazards or defects. Issues that these technicians typically look for include fluid leaks, dents, deflated tires, and cracks. Due to the nature of walkarounds, training usually occurs on-the-job. On-the-job training for this sort of task could pose a potential safety risk. When training on-the-job, instructors have little control over the types of defects available for them to demonstrate. This means that there may exist certain rare issues that technicians are unaware of when they encounter them after training.

It would be impractical to keep a hangar of damaged planes around to train on. The space and cost that would require is unreasonable. A much more feasible solution can be created through virtual reality. Using virtual reality, our software allows United Airlines to store a library of aircraft types as well as defect variations and locations. This would be a significant asset in exposing technicians to potential aircraft issues prior to actually encountering them in operation. By creating a solution using virtual reality, technicians can be trained to spot a much wider variety of defects types, both common and rare. Our virtual reality solution provides the ability to train as well as test and provide feedback to employees.

Functional Specifications

A vital part of ensuring the safety and proper functionality of aircraft during operation is the inspection of the aircraft during a walkaround prior to each flight. The current system for training technicians on walkarounds is on-the-job training. Because there may not be examples of certain defects available during their training period, technicians may have some gaps in their ability to recognize potential issues with an aircraft during a walkaround. An effective solution to this problem is the use of virtual reality to train and test technicians on hazards and defects they might spot during walkarounds.

The project contains a wide variety of scenarios as well as a random mode, which generates a random assortment of defects. The scenarios consist of several different aircraft models in addition to a variety of defect types and locations. The library of aircraft models, scenarios and defects is expandable, allowing a trained developer to add and edit in the future.

The main goals of this project are the training and testing modes. In training mode, the user is taken through the aircraft and shown examples of defects at preset locations. These include text explaining what specifically to look out for and common problems associated with that defect or location. In testing mode, the user moves around the airplane and marks any defects they spot. The user moves around the plane by teleporting to preset locations on a map of the aircraft. After they are finished marking defects, they are presented with a score and a summary of their work that includes the selections they identified correctly, misidentified, and missed altogether. The user then has the opportunity to go back and view the completed scenario. The defects they missed, correctly selected, and incorrectly selected can be reviewed by the user or an instructor. After completing a scenario in testing mode, the user’s ID, scenario number, and score for that scenario are saved. This can be used to track the user’s progress through the scenarios.

A virtual reality solution to training technicians on a walkaround greatly increases the quality of their walkarounds, as they would be more familiar with a wider variety of defects they might encounter. The training would no longer be reliant on defects that they might happen to see during on-the-job training, but rather would consist of training on a nearly extensive library of potential aircraft issues. The software is available for the iPad and Oculus Quest.

Design Specifications

Overview

The Virtual Reality Aircraft Walkaround is a simulation designed to train United Airlines technicians to identify safety issues in various models of planes. Using an Oculus Quest headset or an iPad, trainees can select one of two modes as well as the type of aircraft. The training mode teaches the user by having them walk around and engage with different parts of the plane to view all the possible issues. The testing mode quizzes the user’s knowledge of these issues by placing them in the same simulation, but they must identify the problems themselves. After the user finishes the test, a feedback window will appear and list all the defects on the airplane that were found or missed. The results of the test are saved for instructors to evaluate. In addition to training mode and testing mode, there is a navigation tutorial that teaches the user the controls of the application.

User Interface (UI)

Menu

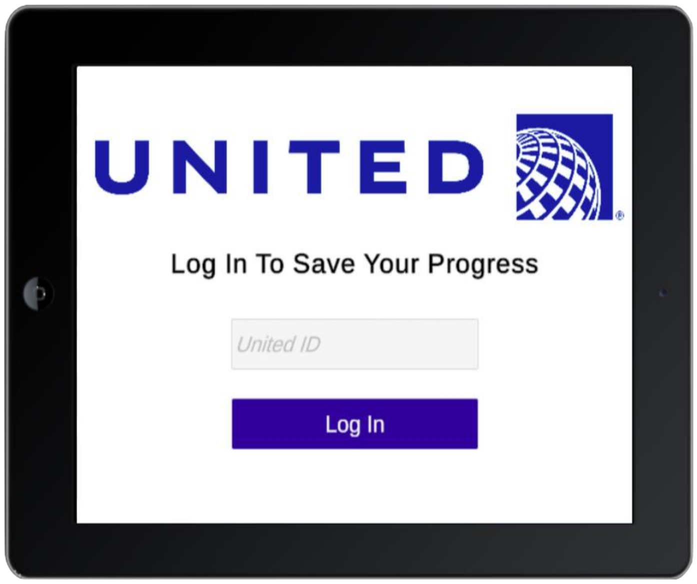


Figure 1: Login Screen

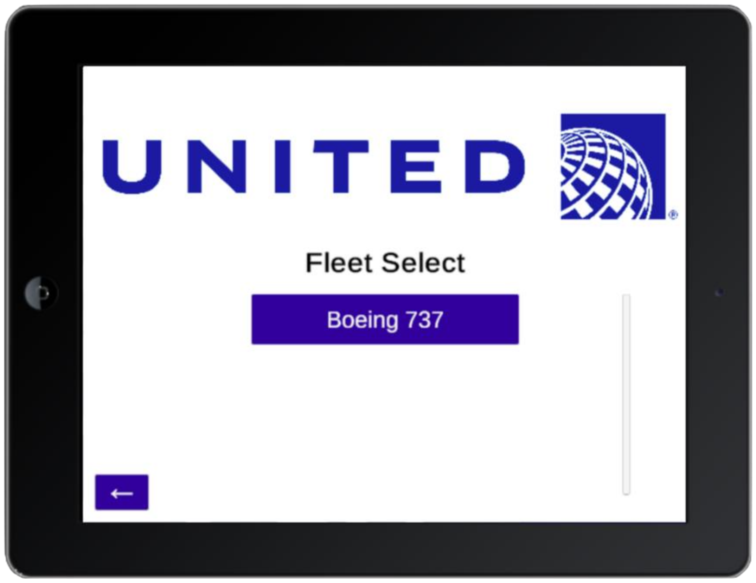


Figure 2: Aircraft selection

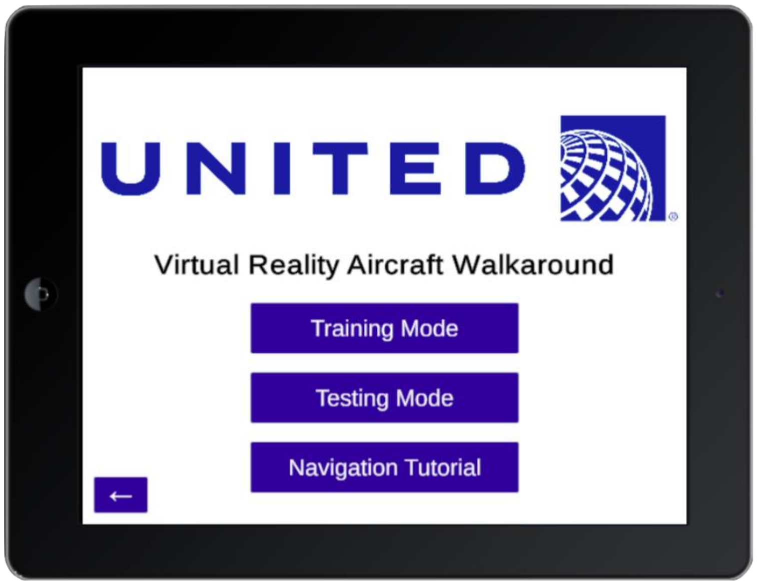


Figure 3: Mode Selection Menu

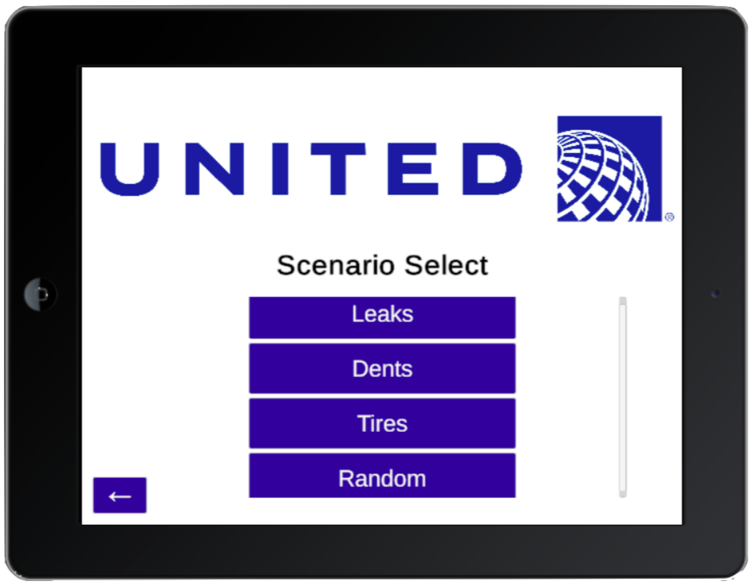


Figure 4: Scenario Select Menu for Testing Mode

As shown in Figure 1, the application starts on the login menu, where the user must input their United ID. From here, the user is able to pick from a list of available aircraft (Figure 2). As seen in Figure 3, the application then moves to a mode selection menu. From here, the user can select Training Mode, Testing Mode, or Navigation Tutorial. If the user selected Testing Mode, a scenario select menu (Figure 4) opens with a list of scenarios with preset defects and an option to have a scenario with random defects.

Aircraft Navigation

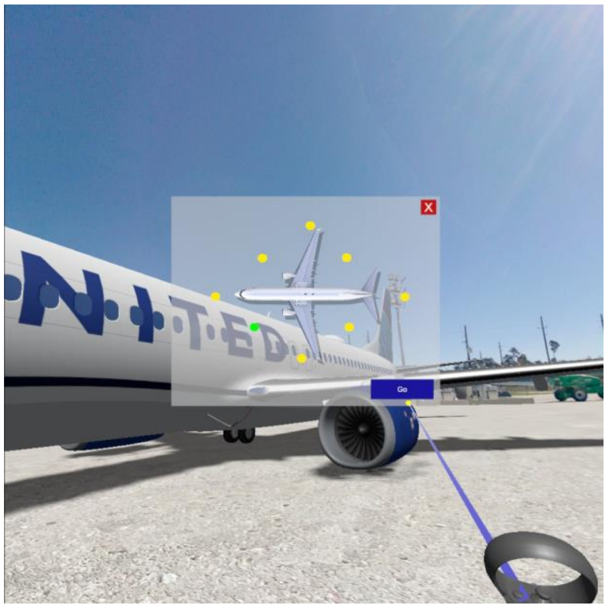


Figure 5: Aircraft Map Navigation

As seen in Figure 5, a popup map of the aircraft can be accessed. This will show all areas of the aircraft that the user can teleport to. Once an area is selected, the point's color is changed to green. The user can then click “Go” to teleport to that area of the aircraft.

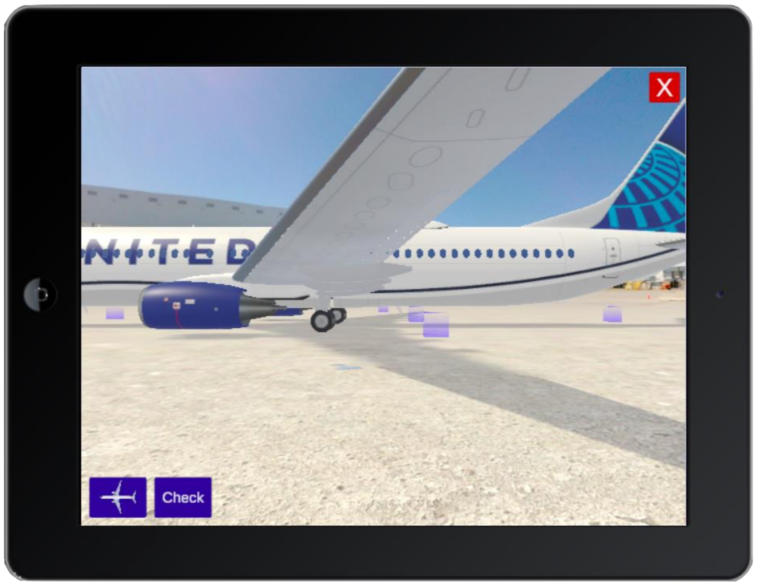


Figure 6: Ground Point Navigation

There are also blue teleportation markers on the ground that correspond to the teleportation points in the popup map. The user is able to click on these teleportation markers to teleport to that area.

Additionally, on the iPad, the user is able to move with a virtual joystick on the screen. The user may also rotate the camera by swiping on the screen and zoom in and out by pinching the screen. On the Oculus, the user moves and rotates the camera directly with the headset.

Training Mode

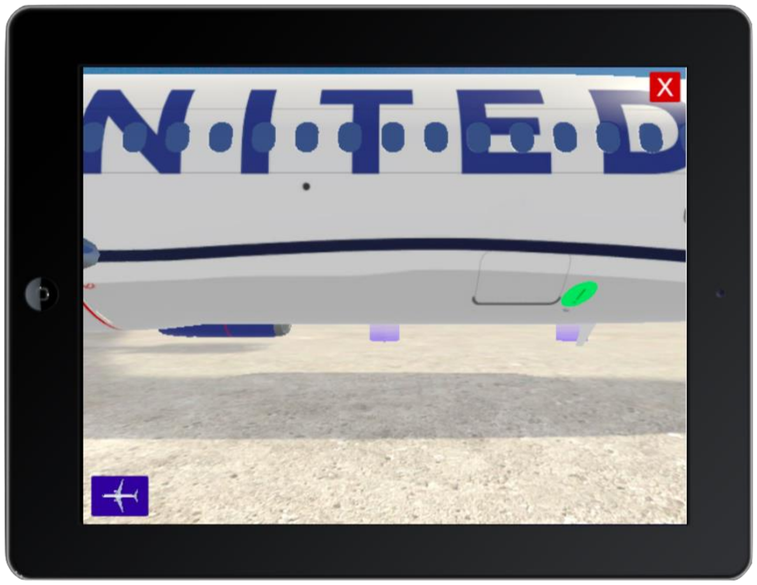
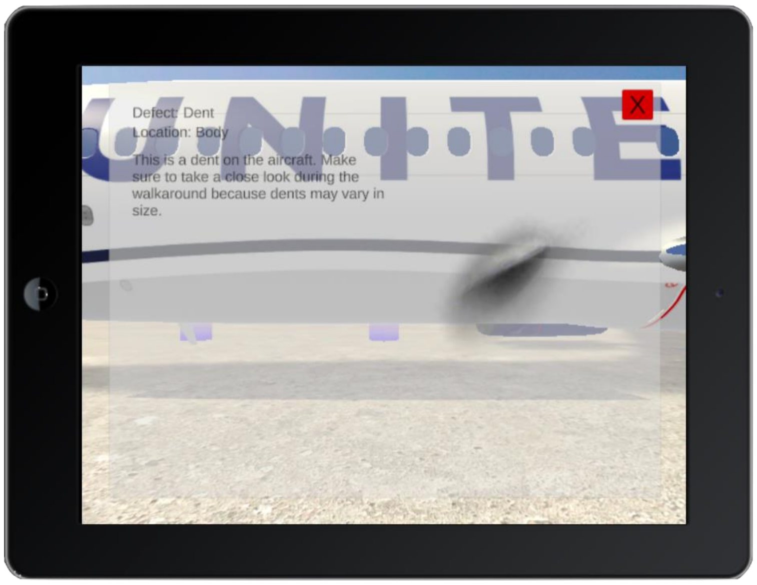


Figure 7: Training Mode Simulation



Defect: Dent
Location: Body
This is a dent on the aircraft. Make sure to take a close look during the walkaround because dents may vary in size.

Figure 8: Training Mode Explanation Text Box

As seen in Figure 7, the simulation is started in Training Mode. All possible plane defects are made visible by an exclamation point. When a user clicks on an exclamation point, a text box appears explaining potential defects in that area that the user should be looking for during a walkaround. An example is shown in Figure 8 above.

Testing Mode

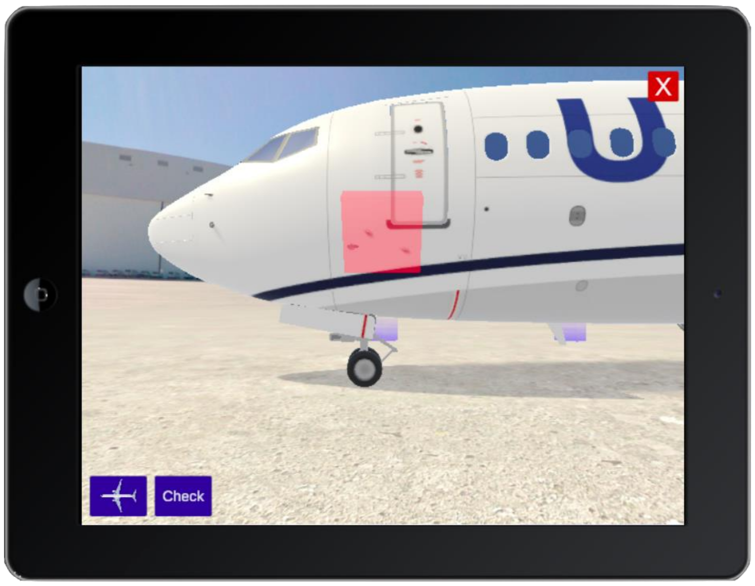
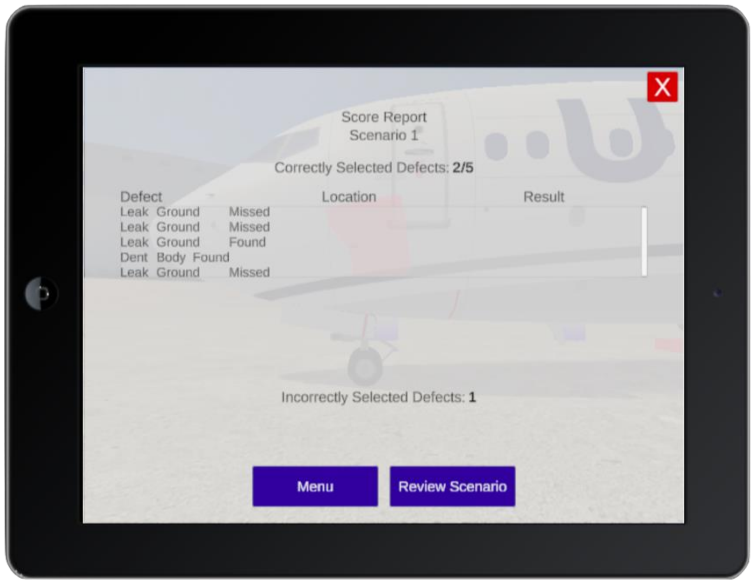


Figure 9: Testing Mode Simulation

Users must identify potential problems and what exactly the issue is by marking defects on the plane. Using an iPad, a user can tap the screen to mark a defect at the tapped location. Using an Oculus Quest, a user can use a button on the controller to mark a defect at the location the controller is aimed at. Marking a defect will cause a visible icon to appear on the aircraft. An example is shown in Figure 9. They are then scored on accuracy and ability to find all given problems.

Feedback Window



```
Score Report
Scenario 1

Correctly Selected Defects: 2/5

Defect                    Location                  Result
Leak  Ground      Missed
Leak  Ground      Missed
Leak  Ground      Found
Dent  Body  Found
Leak  Ground      Missed

Incorrectly Selected Defects: 1
```

Figure 10: Feedback Window for Testing Mode

As seen in Figure 10, a score report is shown after a scenario in testing mode has been completed. The score report includes a list of all correctly selected defects, incorrectly selected defects, and missed defects. The score of each user is stored alongside the scenario number and their user ID for future reference. The user has the ability to review the scenario they just completed, returning to the airplane to view the defects they missed, correctly selected, and incorrectly selected.

Technical Specifications

System Architecture

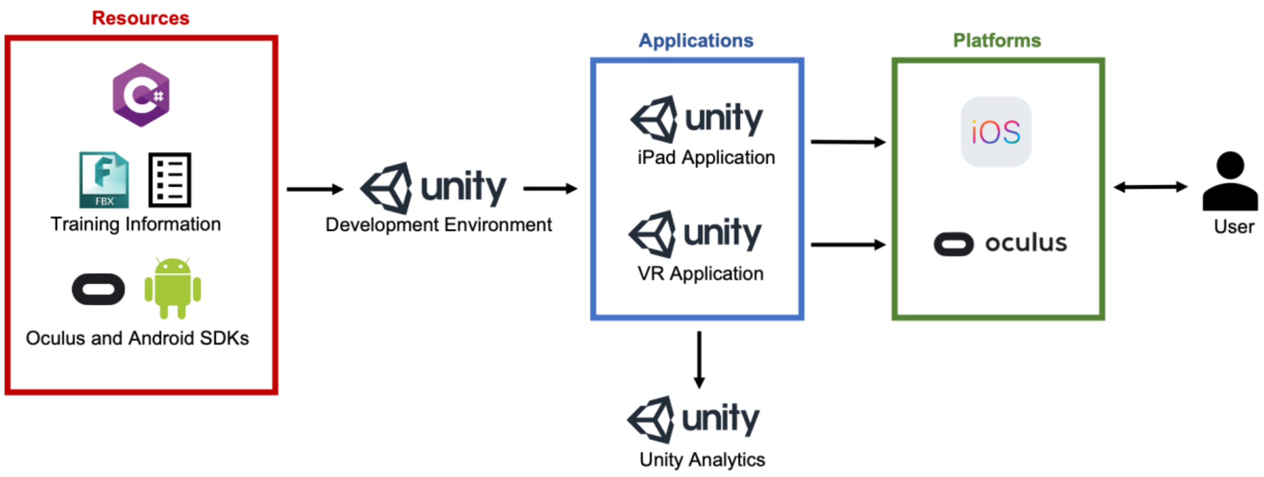


Figure 11: System Architecture Diagram

An end user can access the application using an Oculus Quest or an iPad. The application itself is developed in Unity using C# and the .NET framework and then built for each platform. Building to Oculus Quest requires both Oculus and Android SDKs. While the iPad and VR applications are running and connected to the internet, they send testing data to Unity Analytics.

System Components

Hardware

Oculus Quest

The Oculus Quest is a virtual reality headset created by Oculus VR in 2019. It is fully standalone, meaning the headset itself and its two controllers are completely wireless. It runs on a Qualcomm Snapdragon 835 system on chip, and runs the Android 7.1.1 OS. Trackers in the headset and the two controllers will be used to orient an end user in the application's training simulations.

iPad

The iPad is a tablet first released by Apple Inc. in 2010 and runs iOS. Its three internal gyroscopes and multitouch-capable screen will be used to orient the end user in the application's training simulations.

Development Environments

Unity

Unity, a game engine with both mobile application and VR platform support, will be used for development of the application. The user interface and all model manipulation within the application will be done through Unity. Models are FBX files imported into Unity.

Each defect is represented as either an FBX object or a PNG image. Each defect will be spawned at a position randomly chosen from an array of possible positions for that defect type. For example, if an aircraft is determined to have shoulder wear on one of its tires, a random tire on the aircraft will be replaced with a model of that same tire with the defect present. An aircraft determined to have a dent will have a PNG of a dent placed somewhere on its body, engines, or wings.

Visual Studio 2019

Visual Studio 2019, an IDE from Microsoft, will be used to develop the codebase in C# using .NET and Unity's API.

Additional Software

Xcode

Xcode, an IDE from Apple Inc. will be used for building the application for iOS once it is exported from Unity.

SDKs

There are several SDKs available from Oculus VR for development; Oculus Quest SDK for Unity will be used for building the VR application.

Android 4.4 (SDK 19) and Android 7.1 (SDK 25) will be used for building the VR application as well, since the Oculus Quest uses an Android OS. These, along with the various SDK Tools and Platform Tools required, will all be acquired through Android Studio, an IDE specifically for Android development.

Unity Analytics

Unity Analytics is a data platform that can be used with any Unity project. It can be used to track core usage data, revenue data, and player behavior data as defined by the developers. Data storage is handled by Unity and is accessible only to Unity members that have been given access to it by the project owner. It will be used in this project to track user progress and scores for the various test scenarios.

Risk Analysis

Accuracy of the information and defects within the planes

Difficulty: Low

Description: Since our team members are not highly knowledgeable about potential defects in an aircraft, we need to be careful about making sure the information and defects presented in the application are accurate.

Mitigation: We are keeping in constant contact with the clients and double checking that the information within the application is entirely accurate for the scenarios. We are also scheduled an on-site meeting with the client in order to become more familiar with their aircraft models.

Creating a nearly identical application for both the Oculus Quest and the iPad

Difficulty: Medium

Description: Since the application must be compatible with both the iPad and the Oculus Quest, there will be differences in user controls and interfaces. However, the project must still be functionally identical between the two platforms.

Mitigation: In order to avoid any functional discrepancies between the two versions of the application, both applications are being developed from within the same Unity project, using the same underlying systems with differences only as needed for functionality with the two technologies. The team is actively testing both versions to ensure that the iPad and the Oculus Quest versions are functionally identical.

Different performance levels between the Oculus Quest and the iPad

Difficulty: High

Description: The Oculus Quest is not as powerful as the iPad is. Therefore, problems may occur with being able to run the application on the Oculus Quest.

Mitigation: Development and scaling of the models is primarily done for the Oculus Quest. The iPad version mirrors the Oculus Quest version, with possible differences in order to better fit the iPad controls and screen resolution.

Schedule

Week 1 (1/6 - 1/11)

- Initial team meeting
- Initial client meeting
- Create Slack for communication

Week 2 (1/12 - 1/18)

- Status report presentation
- Initial triage meeting
- Assign roles for each team member
- Download Unity, Xcode, Android SDK
- Obtain an Oculus
- Research technologies that will be used

Week 3 (1/19 - 1/25)

- Rough draft of project plan
- Prototype airplane and defect generation
- Research data storage options
- Learn technologies used

Week 4 (1/26 - 2/1)

- Finish project plan document
- Project plan presentation
- Movement controls on iPad
- Refine airplane and defect generation systems

Week 5 (2/2 - 2/8)

- Teleportation on the iPad
- Start movement controls on Oculus Quest
- Preliminary defect marking system
- Start training mode by creating clickable popup descriptions
- Integrate defect files from client into plane generation structure
- Testing mode basic functionality

Week 6 (2/9 - 2/15)

- Continue movement controls on Oculus Quest
- Start work on laser pointer Oculus controls
- Optimize environment through use of 360-degree images
- Defect marking for iPad

- Refine teleportation on iPad
- On-site visit to Chicago O'Hare February 13-14

Week 7 (2/16 - 2/22)

- Finish laser pointer Oculus controls
- Teleportation on the Oculus Quest
- Defect marking on the Oculus Quest
- Finish alpha presentation
- Practice presentation
- Go over alpha deliverable with the client
- Polish alpha deliverable
- Alpha presentation

Week 8 (2/23 - 2/29)

- Research flashlight lighting
- Start navigation tutorial for Quest
- Refactor defect spawning system
- Refine marking methods
- iPad pinch-to-zoom and joystick

Week 9 (3/8 - 3/14)

- Expand library of defects
- Start audio implementation
- Finish navigation tutorial for Quest and iPad
- Refine testing mode functionality for both platforms
- Research methods for rendering UI in VR

Week 10 (3/15 - 3/21)

- User performance data tracking
- Complete score report information display
- Continue audio implementation
- Update UI render methods in VR

Week 11 (3/22 - 3/28)

- Finish beta presentation
- Go over beta with the client
- Polish and refine the beta deliverable
- Practice presentation

Week 12 (3/29 - 4/4)

- Beta presentation
- Refine and bug fix
- Make sure all aircraft, defects, and scenarios are in the application

Week 13 (4/5 - 4/11)

- Show deliverables to client
- Make any last bug fixes and changes

Week 14 (4/12 - 4/18)

- Create project video

Week 15 (4/19 - 4/25)

- Submit project video
- Submit all deliverables
- Design Day