VIRA | Virtual Insight into Real Athletics

Final Report

Group 33

Advisor

Dr. Diane Rover

Team Members

Katie Perkins	*Team Lead*
Nate Irmiter	*Stakeholder Communicator*
Bailey Righi	*Product Manager*
Willem Paul	*Standardization Manager*
Caroline Rankin	*Meeting Facilitator*

Email

sdmay20-33@iastate.edu

Website

sdmay20-33.sd.ece.iastate.edu

Table of Contents

Table of Figures

1 | Introduction

1.1 | Acknowledgement

We would like to acknowledge the significant help of Joe Hubbard, an athletic trainer at Iowa State who is our primary source of information related to athletic injury examinations. We would also like to thank Dr. Diane Rover for her guidance and expertise throughout the duration of our Senior Design project.

1.2 | Problem and Project Statement

There is nothing as good as hands-on, in-person, real-time experience in performing the tasks of an athletic trainer. When students in Iowa State's athletic training program have made it to the point at which they are able to work in the training room with student athletes, their time there becomes very valuable. However, while they are learning, a full-time athletic trainer must watch them and guide them through what they should do. This makes it very difficult for athletes to get the attention they need when the training room gets busy. On top of that, there are many traumatic injuries that a student will never experience firsthand until they are working full-time in the field.

A virtual reality (VR) training program would allow athletic training students to get the extra experience and feedback they need without having to interrupt or wait for a full-time athletic trainer to finish treating another athlete. We propose a virtual reality program that consists of a set of training modules that walk an athletic training student through injury evaluation and diagnosis scenarios.

1.3 | Requirements

Functional Requirements

Before the Module

- The user can select a user profile when starting the application
- The user can select a specific module
    - Guided Mode modules will focus on special tests
    - Modules will cover traumatic injuries not often seen in training

During the Module

- The user can toggle view modes to see the skin, muscles, or tendons of the virtual limb
- The user can select a muscle or tendon and see its name
- The user can manipulate the limb in a realistic manner
- The application can guide the user through a special test

- The user can exit the module and return to it at any time

After the Module

- The user can see their progress on a particular module over time
- The user can restart the module or return to the menu

Non-Functional Requirements

- The application must respond to user input in real time
- The user must feel comfortable while using the application
  - The application must not induce motion sickness or nausea in most users
- The device must have a battery large enough to operate during a half-hour session

Economic Requirements

- We shall not exceed the funding provided to us by the Department of Electrical and Computer Engineering

Environmental Requirements

- The application must work in real time and interact directly with the user
- The application must accurately interpret the motion of the user's hands and realistically translate that motion to the virtual display
- The user must remain within the Guardian that is free of obstacles and other people while VIRA is in use

1.4 | Intended Users and Uses

VIRA is intended for use by student athletic trainers to practice injury evaluation and diagnosis. These are students who wish to further develop their clinical skills and gain experience working with traumatic injuries they would not typically see as a student worker. Our intended customer is the Iowa State Department of Sports Medicine, as they will be assisting us with module testing and will evaluate our accuracy. Our goal is to create an educational program consisting of modules that will augment students' studies and on-the-job experience.

2 | System Design and Development

2.1 | System Design

Figure 1 – VIRA Component Diagram (right) shows our system component diagram, which we used to guide our design. This diagram shows the different components of the system and how they communicate with one another.

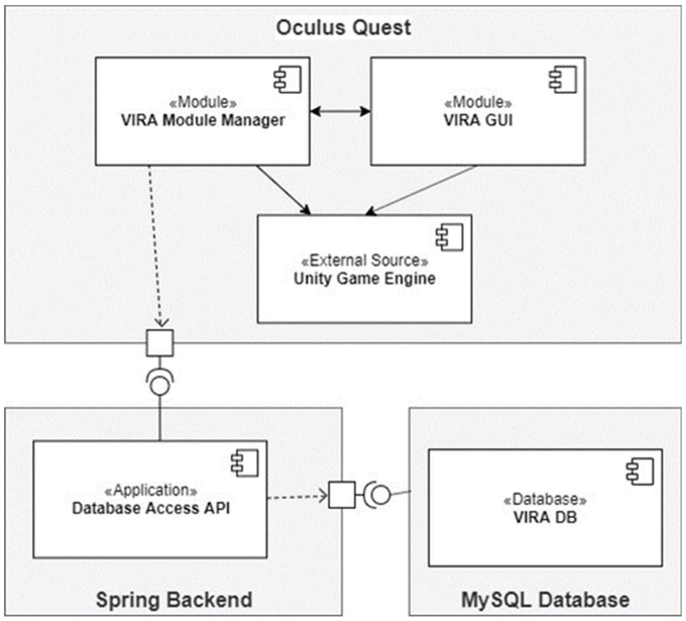

Oculus Quest
«Module» VIRA Module Manager
«Module» VIRA GUI
«External Source» Unity Game Engine
«Application» Database Access API
Spring Backend
«Database» VIRA DB
MySQL Database

Figure 1 – VIRA Component Diagram

The program flow begins with the GUI component. As the user interacts with the application, their input triggers events that are sent to the module manager component. This is our logic layer that runs in the background. It will handle all of the scene changes and program decisions, such as allowing the user to log in, select a module, work through a module, and view their performance. Both the GUI and the module manager components rely on the Unity Game Engine to function. The module manager also interacts with the remote Spring Boot server, which acts as an interface between the application and the database. In order to select a user profile, see user performance, save user performance, or see all modules, interaction with the database is required. Once the data is accessed, it will be cached on the Quest for the duration of the session for efficiency. We chose this method of accessing data because it is much more secure that accessing the database directly from the application and it allows us to complete the necessary calculations for the performance dashboard server-side, decreasing the amount of processing done on the Quest itself. As these user actions are the only ones that require persistent data, communication on the network is minimized.

2.2 | Design Objectives

Our goals for the project were as follows:

1. Create a virtual reality training program that provides student athletic trainers with more experience outside of class and work.
2. Give student athletic trainers realistic simulations of diagnosing injuries without performing real (and critical) examinations.

3. Allow for easy future extension of VIRA by using a modular design.

2.3 | System Constraints

VIRA is primarily designed for stationary, indoor use. The VR headset we have chosen to use—an Oculus Quest—should therefore not be subjected to extreme conditions. As with any VR application, users should be aware of their surroundings at all times. The Oculus Quest has a built-in feature called the "Guardian", a virtual box around the user that alerts them when they step or reach outside of its bounds. This further protects the user and the headset from injury or damage. It does not, however, detect any objects above the user's head, so this should be taken into consideration when using the Oculus Quest.

2.4 | Design Trade-Offs

Originally, we had intended to design VIRA as an augmented reality (AR) application. Unfortunately, we were not able to do so given our financial constraints, so we instead had to create a VR application using the Oculus Quest. We decided upon the Quest as our device since it can be used without being connected to a computer and can utilize some of the same development technologies we had originally planned on using.

2.5 | Diagrams and Design Details

Application Component Diagram

Figure 1 (repeated on the right for convenience) shows a high-level component diagram of our application. Here, we discuss each component in detail.

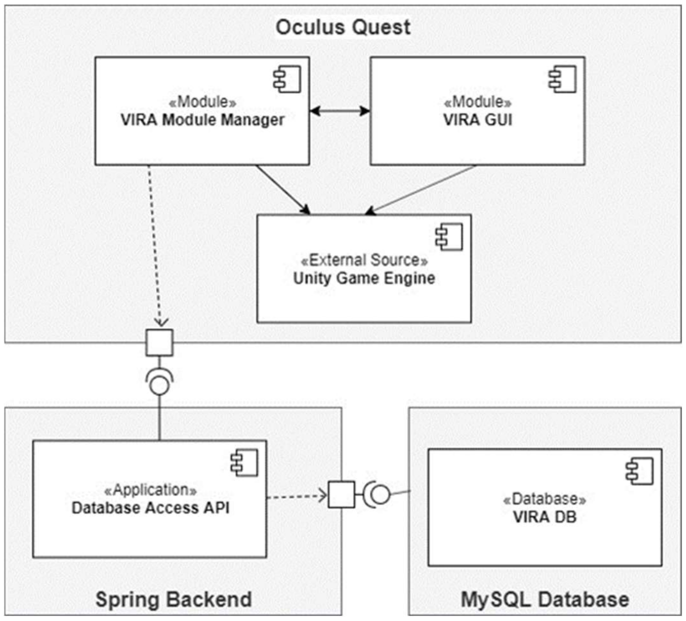


Figure 1 – Application Component Diagram

Module Manager

The Module Manager contains the behind-the scenes logic of the client-side application and interacts with the server. The primary functions of the Module Manager are:

- populating the user profile and guided special test selection screens with data retrieved from the server,
- running the logic of the guided special test module, and
- sending user performance data to the server.

Graphical User Interface (GUI)

The GUI consists of the seven Unity scenes that make up VIRA. These scenes contain the menus and the guided module. Each scene is interactable and is responsible for handling user input, that is, detecting user interaction and correctly assigning events to handlers.

Unity Game Engine

The Unity Game Engine is a powerful, 2D and 3D development software. It is commonly used to create games (hence the name), but is also useful for creating non-game applications, such as VIRA. It contains a built-in physics engine and there are many pre-made assets available. Further discussion can be found in Section 3.1 | Technologies Used.

Data Access API

This API is exposed by our server and allows the application to request data from the database. More detail follows.

Database

Our database is a MySQL database. Further discussion can be found in Section 3.1 | Technologies Used.

Server Component Diagram

Figure 2 – Server Component Diagram shows a component diagram of our Spring Boot server.

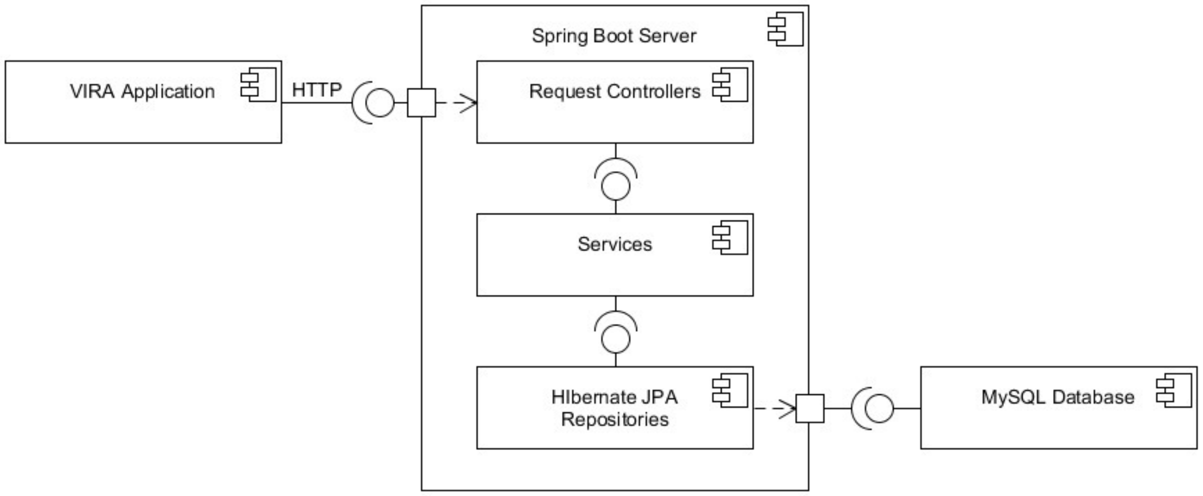


Figure 2 – Server Component Diagram

Request Controllers

To create the API, our server needed to expose a set of possible URIs which the application can use to make requests for data via HTTP. The Controllers are responsible for accepting incoming requests and parsing the URI so that the request can be handled correctly. If the URI is incorrect, the Controllers will automatically return an error message. Otherwise, they will call the correct Services to handle the request.

Services

The Services are responsible for aggregating and transforming data between the form stored in the database and the form required by the application. This helps reduce the computations the application must perform and ensures that we only send relevant data to the application. The Services are organized according to type of data, e.g. special test data, user data, etc. They can call each other to gather more information. For instance, if the application requests the statistics for a user, the User Service needs to communicate with the Special Test Service to collect the correct data.

Hibernate JPA Repositories

Hibernate is a commonly used Object-Relational Mapping (ORM) framework that provides an implementation of the Java persistence API (JPA) to easily manage database create, read, update, and delete (CRUD) operations. A Repository is a JPA representation of database table that can perform both standard CRUD operations as well as more complex SQL queries. These Repositories are used by the Services to retrieve relevant data from and store data in the database.

Database Schema

Figure 3 (next page) shows our MySQL database schema. Each dashed line represents a foreign key constraint. In total, there are 11 tables.

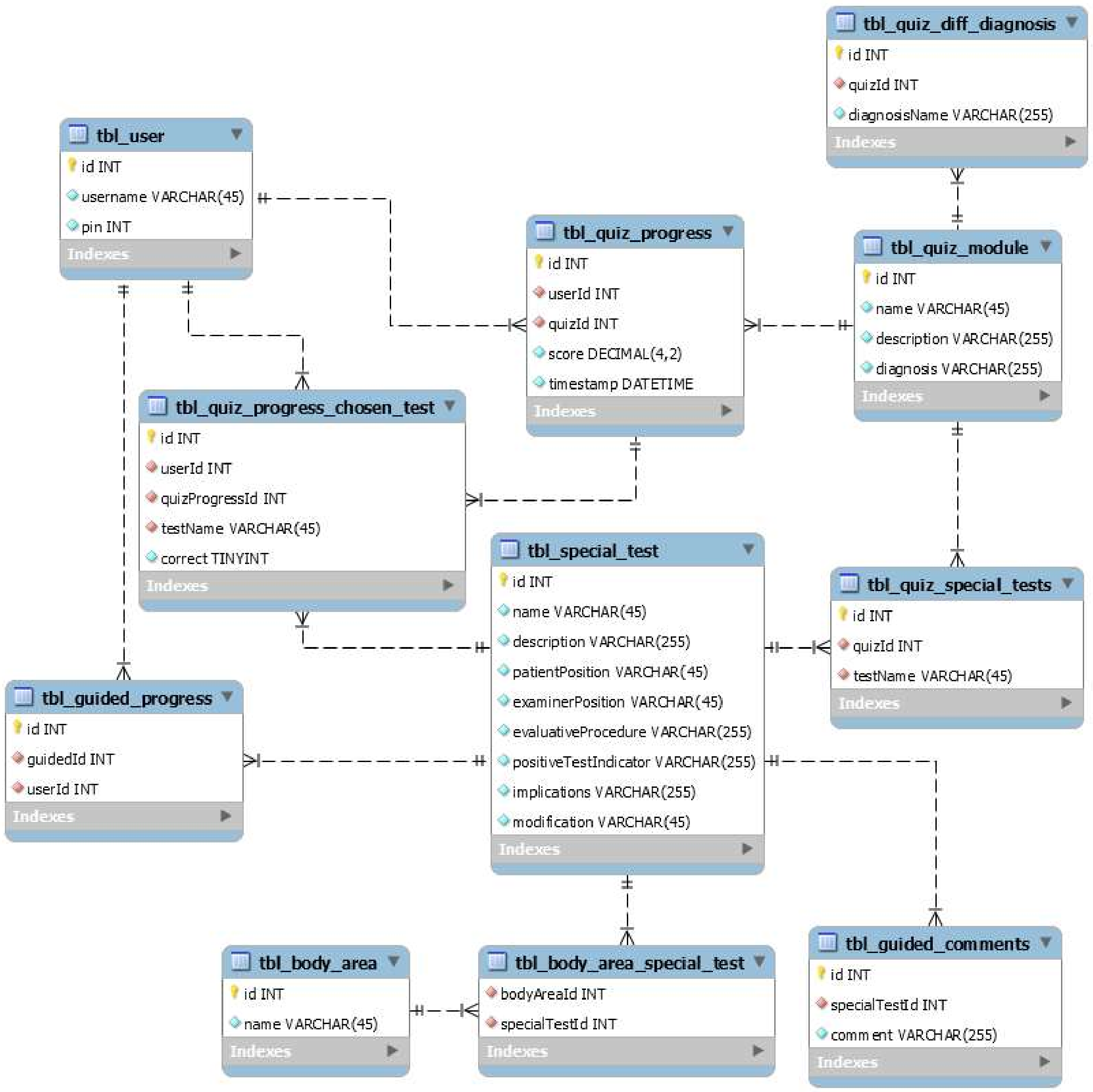


Figure 3 – Database Schema

3 | Implementation

3.1 | Technologies Used

Software

VIRA was created with the Unity Game Engine, utilizing C# for scripting capabilities. The server was written in Java using the Spring Boot and Hibernate frameworks.

The main development tools we used to complete this project were:

- Unity Game Engine
- MySQL Server
- An IDE of choice

Other software we used for the creation of our humanoid models and our application scenes include:

- Blender
- MB-Lab
- Unity Assets

Hardware

We used an Oculus Quest VR headset to implement VIRA.

Rationale

We decided to use the Oculus Quest for our VR hardware because it has the capability to run without being plugged in. This was important because the student trainers must have full range of motion with their arms, which a cord could impede.

We chose the Unity Game Engine for our project because it has a very large userbase and a large collection of tutorials. Since nobody in our project had ever worked with game development software before or had any experience with computer graphics, we wanted software that would allow us to find answers to our questions easily. Unreal is another game engine that we looked into using very briefly, but we decided that Unity was a better fit for our experience level. We also wanted to use software that worked well with our hardware, the Oculus Quest. Oculus has several tutorials, plenty of documentation, and a free Oculus integration plugin, which allowed us to focus solely on VIRA and not on VR compatibility. Unity uses C# scripting to implement more complex details within the game logic, which enabled us to implement our desired functionality.

Another benefit of using Unity is its Asset Store. From the Store, we were able to download scripting and model packages that saved us time. There were three packages that we used in particular. Two of them were used for the prefabs they contained, which had models of a medical table and an office environment (e.g. walls, floor, ceiling, windows, etc.) The third package was fundamental to our guided module, as it provided inverse kinematics (IK) scripts. This is what allows the user to move the humanoid model's arm with real-time animation in the VR environment. After evaluating several different IK assets, we finally decided upon FastIK for its quick setup and easy use.

We chose Blender as our modeling software because it has plugins that allowed us to generate models using existing resources. We tried a couple different methods of model creation; we first tried a program called MakeHuman but ran into issues with clothing moving with the mesh in Unity as well as the objects moving independently of the mesh overall. We settled on a Blender plugin, MB-Lab, because it simplified the modeling process and gave us the ability to snap clothes to the outer mesh of the models, which only required slight modifications instead of major ones. Some of the public resources for MB-Lab are limited since it is still a work in progress, but it still lent itself to our uses better than MakeHuman did since the models were significantly more realistic and ran into fewer issues when it came to manipulation of the mesh objects.

For the server, we decided upon the Spring Boot and Hibernate frameworks because they are some of the most used frameworks for web servers written in Java. They are also easy to use as they hide a lot of the lower-level details behind the API, so implementing our functionality was straightforward and did not take a lot of time. Additionally, the team members are most familiar with Java.

We chose a MySQL server for our database primarily because the team members were most familiar with MySQL. We did not need advanced database functionality and for our purposes, MySQL worked fine.

3.2 | Engineering Standards and Design Practices

Our project is intended to be compliant with the Oculus Quest Virtual Reality Check (VRC) Guidelines [2]. These detail the packaging, audio, performance, functional, and security requirements, among others. All applications submitted to the Oculus Store must comply with these requirements and we intended to develop VIRA as if we were submitting it to the Store.

There are two IEEE standards relevant to our project. The first of these is "P2048.5 – Standard for Virtual Reality and Augmented Reality: Environmental Safety" [3]. Since our project has the goal of helping with injury, the last thing we want is for our users to injure themselves with using VIRA. To mitigate this, we kept safety in mind while designing VIRA's controls so that the users are not required to make any large, sweeping movements. This helps the Oculus Guardian fulfill its task of warning the users before they accidentally hit something or someone nearby.

The second of these standards is “P2048.6 – Standard for Virtual Reality and Augmented Reality: Immersive User Interface” [4]. The whole point of using VR for our project is to be able to complete the tutorials and act in a physical manner that is similar to performing the assessments on a real patient. There is a large difference between keyboard and mouse controls and VR controllers. Because of this, as we designed VIRA’s controls, we ensured that the users would be performing actions in the most intuitive manner possible so that they feel truly immersed in the experience.

There are also several best practices we followed in our development. First and foremost, we needed to ensure that using VIRA was a comfortable experience. Virtual reality can be disorienting and nauseating if not implemented correctly, and we wanted to avoid this. Some primary concerns were:

- limiting extremely bright or extremely dark scenes to reduce eyestrain,
- limiting flashing lights, and
- limiting sudden or fast movements inside the application.

We also followed a modular design in our application and adhered to software best practices, such as code modularity and reuse. Specifically, a script in Unity can be applied to multiple Game Objects (elements in a scene), allowing us to write each script only once. Additionally, a Game Object can be saved as a prefab, allowing us to reuse elements and make only minor changes if need be.

Our user interface is also designed to provide a streamlined, intuitive, and consistent experience. The goal is to enable a user who has never used VIRA to navigate it easily.

4 | Testing

4.1 | Test Plan

The Unity development environment provided us with unit testing and debugging capabilities. It is useful for testing how the module will run on the Quest as well as making it obvious when there are visual bugs. Testing on the Quest depended solely on the device, as we only have one.

We intended to perform user testing with students and trainers within the Athletic Training Program, but due to circumstances caused by the COVID-19 pandemic, we were unable to allow them to test it. Instead, we tested the application on the Quest ourselves and spent time comparing our special test module to several different video examples of the same medical test.

4.2 | Unit Testing

Unit testing was performed continuously while creating scenes in the Unity development environment. Each scene component, such as buttons and other interactable objects, has been tested to ensure that the application flows and functions as intended.

4.3 | Interface Testing

Interface testing was completed by loading scenes onto the Quest to ensure our changes were having the desired effects and not causing issues in VR world space.

We tested our server interface with Postman to ensure that the HTTP requests were received and handled correctly. We also used this to test the server code and make sure it was generating the correct information.

5 | Project and Risk Management

5.1 | Task Decomposition

1. Create user profile selection
    a. Create screen sketches
    b. Develop basic graphics from screen sketches in Unity
    c. Code population of user profiles
        i. Database to server
        ii. Server to application
    d. Code transition from user selection screen to user home screen
        i. Includes interaction with database
2. Create user dashboard screen
    a. Create screen sketches
    b. Develop graphics from screen sketches in Unity
    c. Code transition from selecting an option to its respective scene
3. Create module selection
    a. Create screen sketches
    b. Develop basic graphics from screen sketches in Unity
    c. Code population of module options
        i. Database to server
        ii. Server to application
    d. Code transition from module selection screen to module
4. Develop models for the chosen limb
    a. Determine what model capabilities in VR should be
        i. Determine which technologies provide these capabilities
        ii. Test multiple technologies/plugins/programs
    b. Fully outfit and rig models to ensure VR compatibility
        i. Create "ghost limb" to act as a guide for the user
    c. Import models into Unity and recolor them for realism
        i. Apply scripts to make models interactable
        ii. Correctly position models
    d. Animate the chosen limb
5. Create guided special test module
    a. Establish narration/guided content with help from athletic trainer
    b. Establish statistics to collect for performance dashboard
    c. Create screen sketches
    d. Develop basic graphics from screen sketches in Unity
    e. Code ability for user to interact with and move a limb

   f. Code text and visuals for user guidance
      i. Limb movement indicators to guide user to move the limb in a certain way
      ii. Button selection for next step
   g. Code detection that the user has followed guidance correctly
      i. Detect if the user has moved the limb to the correct position as indicated (if applicable)
      ii. Code redirect in case the user has done something wrong (may include more specific guidance or more explanations)
   h. Code transition from module completion to module selection screen
      i. Switch scenes
      ii. Send user statistics to server
      iii. Store user statistics in the database
6. Create user performance dashboard
   a. Establish which statistics will be used or not based on the module type
   b. Create screen sketches
   c. Develop basic graphics from screen sketches in Unity
   d. Code user-specific data retrieval from database
   e. Code necessary calculations to display the correct statistics
   f. Code statistics display
7. Set up database
   a. Design database architecture
      i. Decide on tables and columns
   b. Create database/set up MySQL server
   c. Populate with test data
8. Establish connection between VIRA and database
   a. Create Spring Boot backend
      i. Create Controllers to handle incoming HTTP requests
      ii. Create Services to gather and transform data
   b. Establish connection from VIRA to server
      i. Request information from server via HTTP
      ii. Populate application objects with information from server
      iii. Send user data to server via HTTP
   c. Establish connection from server to database
      i. Configure Hibernate JPA repositories to perform CRUD operations on the database
9. Enhance graphics quality in Unity
   a. Improve humanoid model graphics
   b. Establish a standard look for the UI

5.2 | Project Schedule

The following pages show a Gantt chart of our project progress. Due to the nature of Gantt charts, the formatting may not be ideal.

TASK NAME	START DATE	END DATE	DURATION (WEEKS)	October				November				December				January				February			
				1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Create User Profile Selection																							
Create screen sketches	10/9	11/3	4		🟪	🟪	🟪	🟪															
Develop basic graphics from screen sketches in Unity	2/3	2/4	1																	🟪			
Code population of user profiles	2/17	3/10	4																		🟪	🟪	🟪
Code transition from selection of user screen to user home screen	4/2	4/23	3																				
Create User Dashboard Screen																							
Create screen sketches	10/9	11/3	4		🟪	🟪	🟪	🟪															
Develop basic graphics from screen sketches in Unity	2/3	2/4	1																	🟪			
Code transition from selecting an option to its respective scene	2/17	3/10	4																		🟪	🟪	🟪
Create Guided Mode Special Test Module Selection Screen																							
Create screen sketches	10/9	11/3	4		🟪	🟪	🟪	🟪															
Develop basic graphics from screen sketches in Unity	2/3	2/4	1																	🟪			
Code population of module options	2/3	3/3	5																	🟪			
Code transition from selection of module screen to starting the module	2/3	2/18	3																	🟪	🟪	🟪	
Develop Models for the Chosen Limb(s)																							
Determine what we want the model to be capable of doing in VR	1/21	1/27	1													🟪							
Determine what technologies may get us models with the capabilities desired	1/21	1/27	1													🟪							
Test out multiple technologies/plugins to see if they would be a method of best fit	2/18	2/23	1																			🟪	
Fully outfit the models so they are rigged up and ready for VR manipulation	2/18	3/10	3																			🟪	🟪
Import models into Unity3D and recolor them so they look realistic	2/18	3/10	3																			🟪	🟪
Animate the chosen limb	2/18	3/10	3																			🟪	🟪
Create Guided Special Test Module																							
Establish narration/guided content with help from athletic trainer	3/23	3/26	1																				
Establish statistics to be collected for performance dashboard	3/23	3/26	1																				
Create screen sketches	10/9	11/3	4		🟪	🟪	🟪	🟪															
Develop basic graphics from screen sketches in Unity	2/3	2/3	1																	🟪			
Code ability for user to interact with and move a limb	1/27	2/17	3																🟪	🟪	🟪		
Code text and other visuals for user guidance	4/23	5/1	2																				
Code detection that user has followed guidance correctly	4/23	5/1	2																				
Code transition from module completion to module selection screen	4/23	5/1	2																				

TASK NAME	START DATE	END DATE	DURATION (WEEKS)	March				April				May			
				1	2	3	4	1	2	3	4	1	2	3	4
Create User Profile Selection															
Create screen sketches	10/9	11/3	4												
Develop basic graphics from screen sketches in Unity	2/3	2/4	1												
Code population of user profiles	2/17	3/10	4	🟪											
Code transition from selection of user screen to user home screen	4/2	4/23	3					🟪	🟪	🟪					
Create User Dashboard Screen															
Create screen sketches	10/9	11/3	4												
Develop basic graphics from screen sketches in Unity	2/3	2/4	1												
Code transition from selecting an option to its respective scene	2/17	3/10	4	🟪											
Create Guided Mode Special Test Module Selection Screen															
Create screen sketches	10/9	11/3	4												
Develop basic graphics from screen sketches in Unity	2/3	2/4	1												
Code population of module options	2/3	3/3	5												
Code transition from selection of module screen to starting the module	2/3	2/18	3												
Develop Models for the Chosen Limb(s)															
Determine what we want the model to be capable of doing in VR	1/21	1/27	1												
Determine what technologies may get us models with the capabilities desired	1/21	1/27	1												
Test out multiple technologies/plugins to see if they would be a method of best fit	2/18	2/23	1												
Fully outfit the models so they are rigged up and ready for VR manipulation	2/18	3/10	3	🟪											
Import models into Unity3D and recolor them so they look realistic	2/18	3/10	3	🟪											
Animate the chosen limb	2/18	3/10	3	🟪											
Create Guided Special Test Module															
Establish narration/guided content with help from athletic trainer	3/23	3/26	1				🟪								
Establish statistics to be collected for performance dashboard	3/23	3/26	1				🟪								
Create screen sketches	10/9	11/3	4												
Develop basic graphics from screen sketches in Unity	2/3	2/3	1												
Code ability for user to interact with and move a limb	1/27	2/17	3												
Code text and other visuals for user guidance	4/23	5/1	2								🟪	🟪			
Code detection that user has followed guidance correctly	4/23	5/1	2								🟪	🟪			
Code transition from module completion to module selection screen	4/23	5/1	2								🟪	🟪			

TASK NAME	START DATE	END DATE	DURATION (WEEKS)	October				November				December				January				February			
				1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Create user performance dashboard scene																							
Establish which statistics to display for each mode	3/3	3/3	1																				
Create screen sketches	2/4	2/17	2																	🟪	🟪		
Develop basic graphics from screen sketches in Unity	2/3	2/18	3																	🟪	🟪	🟪	
Code data retrieval from database for current user	2/18	3/2	2																				🟪
Code calculations needed to display the right statistics	4/23	5/1	2																				
Display statistics in visually pleasing way that is helpful to user	4/23	5/1	2																				
Set-Up Database																							
Design database architecture - decide on tables and columns	10/29	1/21	12				🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪	🟪					
Create MySQL database and set up on server	1/21	1/27	1															🟪					
Populate with test data	1/21	1/30	2															🟪	🟪				
Establish connection between VIRA and database																							
Create a Spring Boot backend	1/21	2/16	4															🟪	🟪	🟪	🟪		
Establish connection from VIRA to server	2/18	2/18	1																			🟪	
Establish connection from server to database	1/21	2/16	4															🟪	🟪	🟪	🟪		
Enhance graphics quality in Unity																							
Improve human models	3/2	3/10	2																				
Establish a standard look for UI	4/23	5/1	2																				

TASK NAME	START DATE	END DATE	DURATION (WEEKS)	March				April				May			
				1	2	3	4	1	2	3	4	1	2	3	4
Create user performance dashboard scene															
Establish which statistics to display for each mode	3/3	3/3	1	🟪											
Create screen sketches	2/4	2/17	2												
Develop basic graphics from screen sketches in Unity	2/3	2/18	3												
Code data retrieval from database for current user	2/18	3/2	2	🟪											
Code calculations needed to display the right statistics	4/23	5/1	2								🟪	🟪			
Display statistics in visually pleasing way that is helpful to user	4/23	5/1	2								🟪	🟪			
Set-Up Database															
Design database architecture - decide on tables and columns	10/29	1/21	12												
Create MySQL database and set up on server	1/21	1/27	1												
Populate with test data	1/21	1/30	2												
Establish connection between VIRA and database															
Create a Spring Boot backend	1/21	2/16	4												
Establish connection from VIRA to server	2/18	2/18	1												
Establish connection from server to database	1/21	2/16	4												
Enhance graphics quality in Unity															
Improve human models	3/2	3/10	2	🟪	🟪										

5.3 | Roles and Responsibilities

Katie

As team lead, Katie planned out our timeline and created our initial tasks. As time went on, she created new tasks as needed and helped distribute tasks to the appropriate team members. She helped the team prioritize their work to ensure everyone had work to do without being overwhelmed. In terms of project work, Katie focused on the guided special test module, which included:

- researching and implementing what was needed to allow user manipulation of the virtual athlete (inverse kinematics, C# scripting, configurable joint components),
- animating the "ghost arms" used for user guidance,
- designing and implementing the graphics for the virtual training room, and
- making all user interaction with the human model VR-compatible.

Due to the COVID-19 situation in the latter half of the semester, she also was the user tester for all the screens on the Oculus Quest.

Nate

As the communications representative for the team, Nate was responsible for messaging advisors and other organizations to communicate on behalf of the team. For the duration of the project, he helped determine which modeling technologies we would be using and created, rigged, and tested these models until they were functioning properly. This involved learning complicated modeling software and attempting multiple techniques until one was found that worked sufficiently. He also assisted in setting up the database and server code with Willem.

Bailey

As product manager, Bailey monitored the progress of our application and checked it against our planned timeline. She organized sprints in GitLab and set up boards to assign and track project tasks. Her programming focus was on C# scripts in Unity, from server requests to UX. She also served as a liaison with athletic trainers by setting up group meetings with them and by getting their feedback on UI details.

Willem

As standardization manager, Willem's was responsible for ensuring that our application had a consistent appearance. He was responsible for creating the general UI look and feel (with help from Caroline) and for designing the color and font scheme we used. In addition, he worked with Nate to set up the database and was the primary developer of the server code. He also worked to ensure that the team's work followed similar formatting to make it easier for every group member to understand each other's work.

Caroline

As the meeting facilitator, Caroline helped with organizing the team's schedule with advisor meetings, class assignments, and other team events. During the project, Caroline worked largely on UI design and implementation of VIRA in Unity with C# to create some scenes of the app. As part of that work, she worked closely with Bailey on implementation and with Willem on design. She also used server requests to pull information into their respective scenes.

5.4 | Risks and Mitigation

There are a few different risks inherent to our project. First, because we were forced to switch from AR to VR, there was the risk that VIRA would not function as we initially envisioned. This meant that we had to compromise on some implementation decisions to make VIRA work properly in VR. Similarly, VIRA is designed for athletic training students and thus requires medical and anatomical accuracy that none of us are too familiar with. In order to mitigate the risk of our module not working as the client intends, we worked closely with them to ensure that our application was accurate and beneficial.

6 | Conclusion

6.1 | Lessons Learned

Throughout the year, we learned a lot about working on a project and working as a team. One of the most important things we learned was that time management is essential for an experimental project like this one, and we had to figure out how to balance that with our other classes. Eventually, we ended up having to prioritize certain features over others as the amount of research and prototyping time needed was far more than we anticipated. While we are disappointed that we couldn't accomplish the product we had originally envisioned, after going through all of the hard work to complete what we could, we are very proud of where we ended up. We also had the new experience of having to work remotely as a team. We had to learn how to communicate effectively and efficiently either through GroupMe or WebEx, as well as how to make a plan for switching over to working completely remotely.

We also learned several new things about computer graphics and developing a virtual reality application. Coding for VR allowed us to think more about how we could create an environment that was comfortable for our users while still having an intuitive UI design. This is something that we would not have given much thought if we were making an application for normal computers. We also were able to work with VR hardware for this project, which gave us a unique experience as Software Engineering students, since we typically do not have to go through the process of packaging software and transferring it to hardware in non-Computer Engineering classes. Finally, we learned several new graphics concepts while implementing our project. Below are the ones we found to be the most challenging and interesting:

- 3D human modeling, mesh creation, and rigging
- Object hierarchies, creation, and manipulation
- Event handling
- Animation of 3D models in Unity
- Forward and inverse kinematics
- Procedural animation

6.2 | Closing Remarks

Our ultimate goals were to:

1. develop at least one complete special test module in VIRA,
2. design a modular product to allow for easy extensibility, and
3. learn about VR development.

We feel that we accomplished some of our goals and got the framework set up for future development, but in the end, we did not have time to finish what we had originally planned. Unfortunately, due to the COVID-19 pandemic, we lost about two weeks of work, and that, combined with other difficulties with the project, meant we had to cut out some of our end goals. All in all, we were able to develop a complete VIRA guided module for the Neer Impingement test. We were able to create this product in a way that was modular and would make future modules easy to add on; we simply ran out of time to implement more. Throughout the course of two semesters, we feel that we have learned a lot about VR development and what tools are publicly available for programmers. If we had to go back and do our project again from the start, it would be much smoother because we would have a better idea of which technologies worked and which were not all that useful (which was hard to know without spending a lot of time troubleshooting in many circumstances).

We also believe it is worth noting the division of time between research and implementation in our project: there was a very heavy portion of time devoted entirely to research and prototyping. This was due to a few reasons. The first is that we were not able to apply many skills we had learned in previous classes to the majority of our project, with the server-side code and database being the main exceptions. The entire team was new to virtual reality development, which meant that we had no experience with Unity, Blender, MakeHuman, and MB-Lab. All these software tools are robust systems that often take years to fully master, and we were working with a timeline of two semesters. Lots of time went into just learning how to use the software itself, not to mention learning the common vocabulary of computer graphics or how to apply the mathematical concepts that the field of computer graphics relies on. Not only that, but the functionality that we wanted for our guided module—determining the animation of a limb based on the user's actions—is not at all common in game development. It took a month of researching just to find the vocabulary terms we needed—inverse kinematics and procedural animation. So while we may not have accomplished everything that we had wanted to when we proposed this project, we feel that what we were able to learn is just as valuable as the final product that we delivered.

Appendices

Appendix A | Operation Manual

Using the Oculus Quest

The first step to using VIRA is to be familiar with the Oculus Quest. There are three components: the headset and two controllers (see *Figure 4* - Oculus Quest Headset and Controllers). The Oculus Quest User Guide can help with properly setting up the Quest and with getting familiar with the controls. The Quest is shipped with a demo application in which you can practice using the controls.

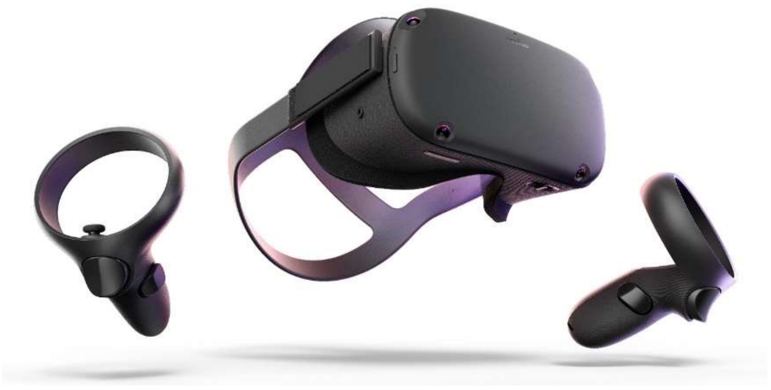


Figure 4 - Oculus Quest Headset and Controllers

Using VIRA

To start the VIRA application, navigate to the Oculus Home. This can be accomplished by pressing the Home button on the right controller. From this screen, navigate to the list of applications (on the left) and select "VIRA".

The application will load and display the main menu (see *Figure 5*). From here, select "Select User". This will navigate to the user profile selection screen (see *Figure 6*). A list of usernames will be presented; select the username associated with your profile. This will take you to your User Dashboard. From here, you can

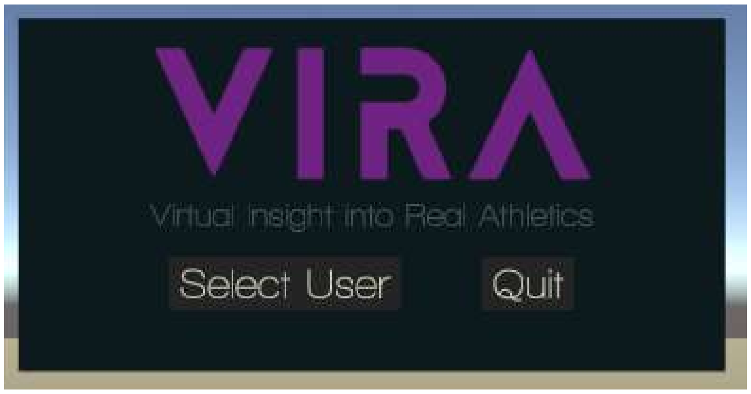


Figure 5 – Main Menu

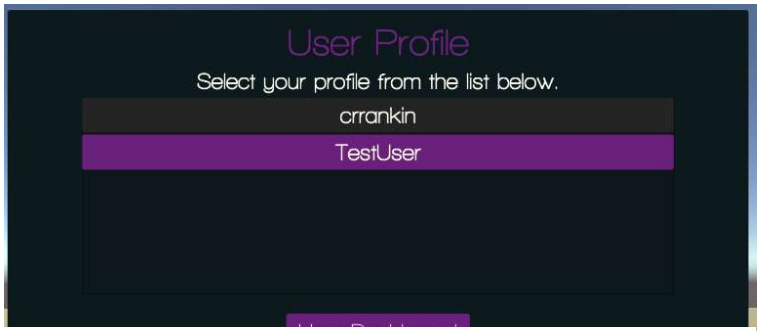
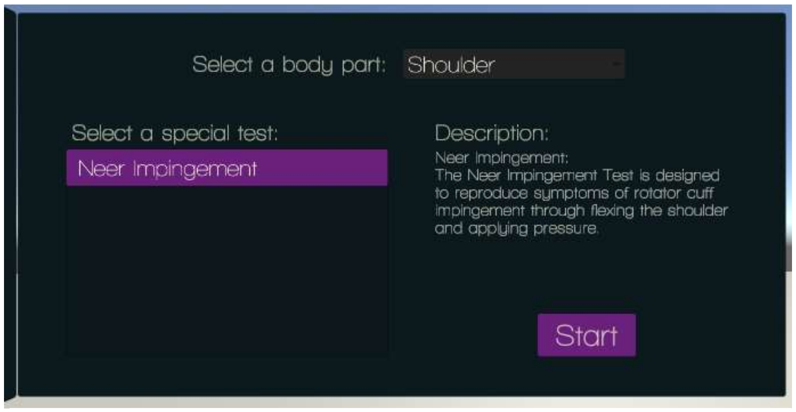


Figure 6 - User Profile Selection Screen

navigate to a guided special test, a quiz, or your performance dashboard.

Special Tests

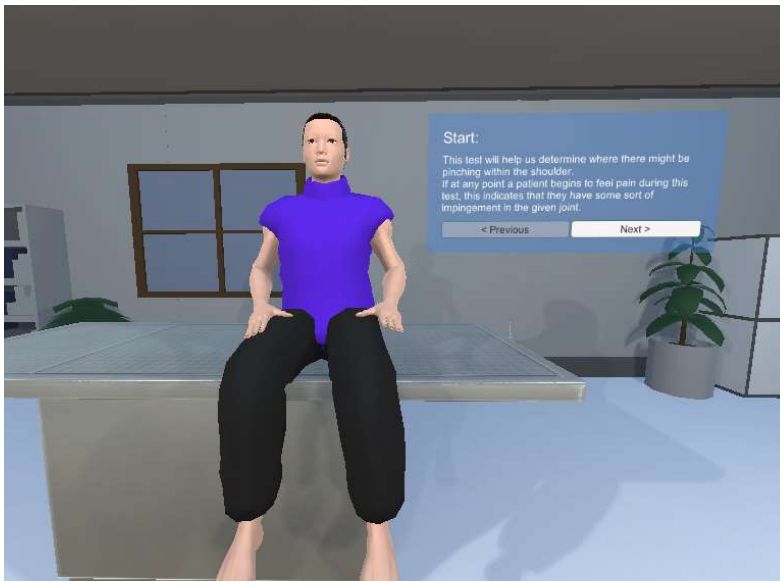
VIRA is a set of guided special tests to help you practice various examinations to determine injuries. From the guided special test selection screen (see *Figure 7*), you can select a body area to focus on. A list of tests involving that area of the body will be displayed. Selecting a test from this list will display a short description of the test. When you have found the test you wish to practice, select "Start".



Neer Impingement:
The Neer Impingement Test is designed to reproduce symptoms of rotator cuff impingement through flexing the shoulder and applying pressure.

Figure 7 – Special Test Selection Screen

When the special test loads, you will see the patient on a table (see *Figure 8*). Next to the patient are the instructions that will guide you through the special test. To complete the module, simply follow the instructions. If you need to reread a step, you can navigate back and forth between the steps.



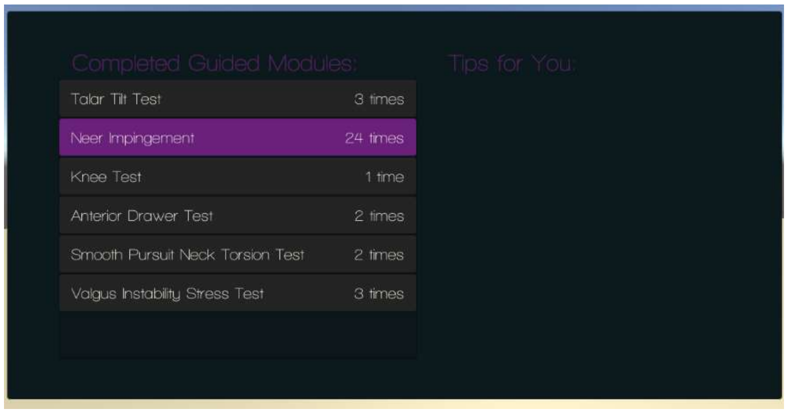
This test will help us determine where there might be pinching within the shoulder.
If at any point a patient begins to feel pain during this test, this indicates that they have some sort of impingement in the given joint.

Figure 8 – Special Test Module

When you are finished with the test, you will be taken back to the User Dashboard and your progress will be automatically recorded.

Performance Dashboard

In the Performance Dashboard (see *Figure 9* – Performance Dashboard, next page), you can see your performance data. This includes statistics such as a list of which tests you have practiced, how many times you have completed each test, and your most- and least-


Completed Guided Modules:

Tips for You:

Talar Tilt Test	3 times
Neer Impingement	24 times
Knee Test	1 time
Anterior Drawer Test	2 times
Smooth Pursuit Neck Torsion Test	2 times
Valgus Instability Stress Test	3 times

Figure 9 – Performance Dashboard

recognized tests. You can return to the User Dashboard at any time by looking at the panel with three large buttons and selecting "User Dashboard" and then "Go!".

Notes

You will need to be on the Iowa State campus to use VIRA. The server requires a connection to the Iowa State network and the Oculus Quest does not have the capability to use a virtual private network (VPN) to access the server from off campus.

If at any time during use you begin to feel nauseous or dizzy, stop immediately and remove the headset. Virtual reality can be disorienting for some and can cause nausea and headaches. Please take time to adjust if you are unfamiliar with VR. We have tried our best to make VIRA as comfortable an experience as possible, but we cannot guarantee complete user comfort.

Appendix B | Other Considerations

Model Development

The process of developing realistic humanoid models that would allow us to manipulate them as we wanted was long and led to many dead ends. This appendix contains some examples of our various iterations we tested before finally reaching our final result.

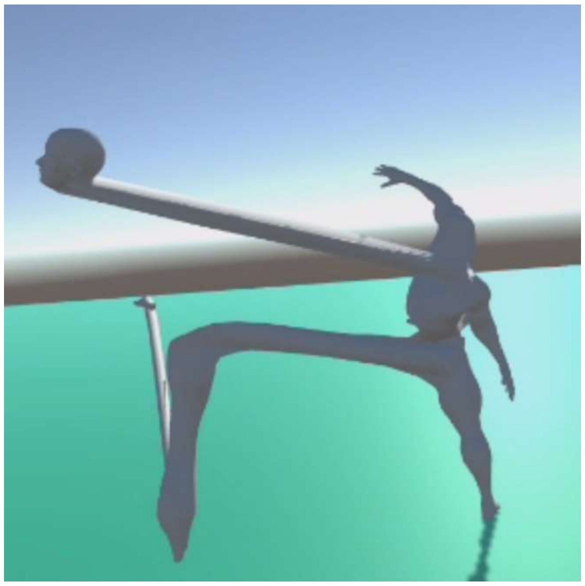


Figure 10 – We had some fun messing with the models once we figured out how to make them

Figure 11 – The models looked alright, but were naked and gray

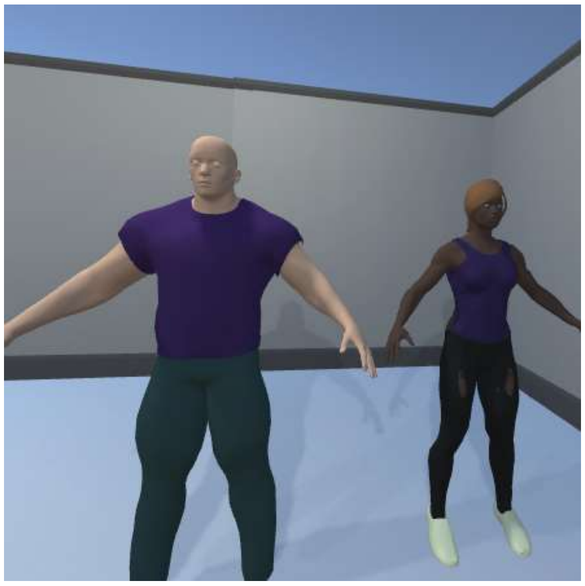


Figure 12 – Using MB-Lab, we managed to clothe and color the models, but there were still some issues

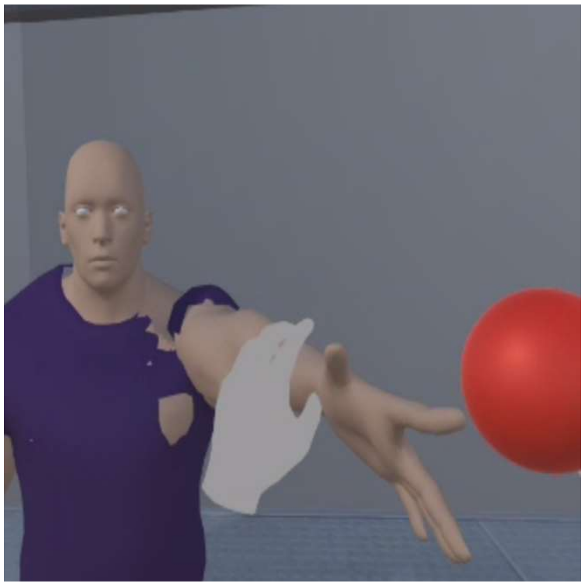


Figure 13 – Our first attempt at applying inverse kinematics to a model led to some interesting results

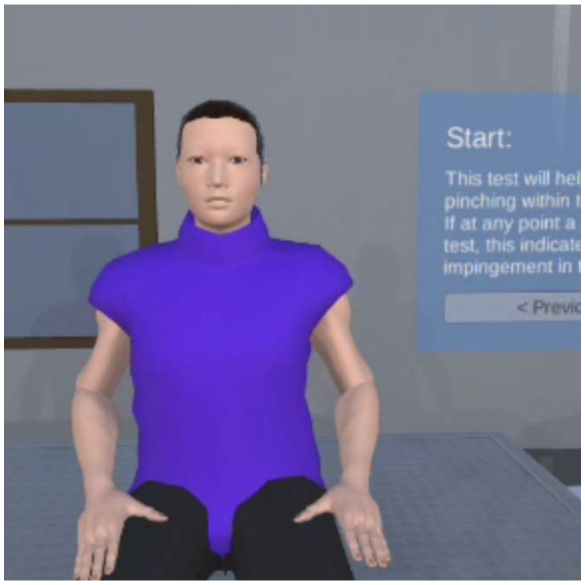


Figure 14 – A finished model

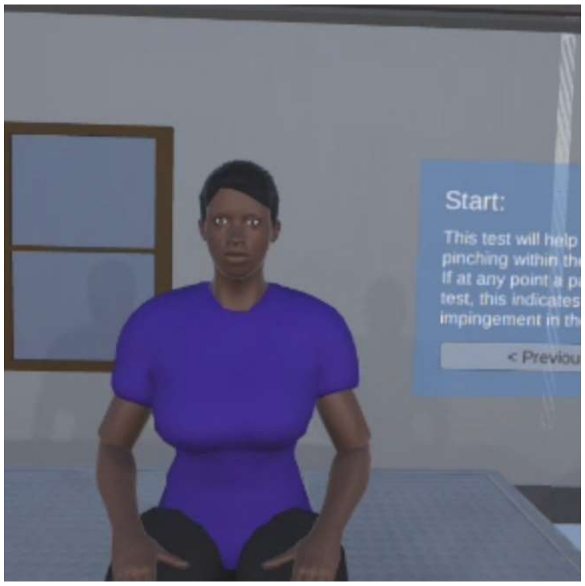


Figure 15 – A finished model