**Sport Science Physiology Simulation Design Document**

# High Concept

Application designed to be used as a teaching aid that is programmed to simulate the physiological responses of the human body when using sports science equipment.

# Team Roles

*Producer*

* Emily Salisbury

*Code*

* Sam Hunt
* Alex Marshall
* Conor Burke

*Art*

* Georgie Phimister

*Design*

* Bethany White
* Alex Vaughan

# Notes & Considerations

* Accessibility for students (available online) - eligible for those unfamiliar with games
* Change variables mid-simulation to see changes take effect
* Optional safety warnings/alarms
* Resizeable for different resolutions
* Text size options - accessibility

# Application Overview

###### Description

A 2D virtual lab to be used as a teaching aid for Sports Science students. The application will allow the user to select between 3 modules, pulmonary, cardiovascular and metabolic, as well as giving the option to create a custom module. Each module will have basic and advanced templates except the custom module. Once the module and template have been chosen, the user will be able to view a simulated human, with limited customization options available. The user can change what equipment the simulated human will be using at any time and track the variables from the chosen module as they change over time. Additionally, the user will be able to set certain variables themselves that will affect the variables being output by the simulation. The user will have displayed the physiological data being output from the simulated human in the form of figures (graphs), numbers or as both.

###### Objectives

* Accurately simulate physiological responses to the human body according to variables set either by the application (Random or Default) or the user.
* View the relevant corresponding data to these simulations in table/graph form.
* Allow the user to change which variables are being viewed, and to change the values of certain variables mid-simulation, such as speed, duration, etc.
* Give the option of at least 3 pieces of equipment including bike, treadmill and rower

###### Stretch Goal Objectives

* Exam mode/Functionality for assessment purposes (needs further discussion with client)
* Safety warnings/alarms at dangerous levels
* Database for a login function
* Additional modules and more of the equipment being visually represented
* Animation of the character using the equipment
* Sound effects for the animation and UI

###### Target Audience

Sports Science Students & Faculty

###### Platform

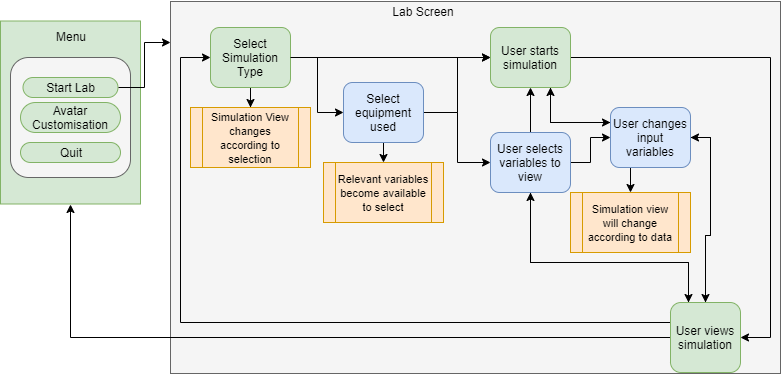
Target is to develop a standalone application (.exe) for PC.

* MAC Compatibility?

# Application Mechanics

User will be able to select individual monitoring equipment or ‘modules’ (defined by a particular group of variables, eg cardiovascular or pulmonary variables). This will enable them to be able to select exercise equipment as seen on the screen.

Modules include all of the relevant variables to that type of data. Users will be able to select a custom modules to view data from different modules at the same time but will never be able to view all of the variables at once as there is a limit. Once a module has been selected, the user will be able to remove specific variables from being viewed if they wish.

Gameflow

# Controls

Mouse - point and click

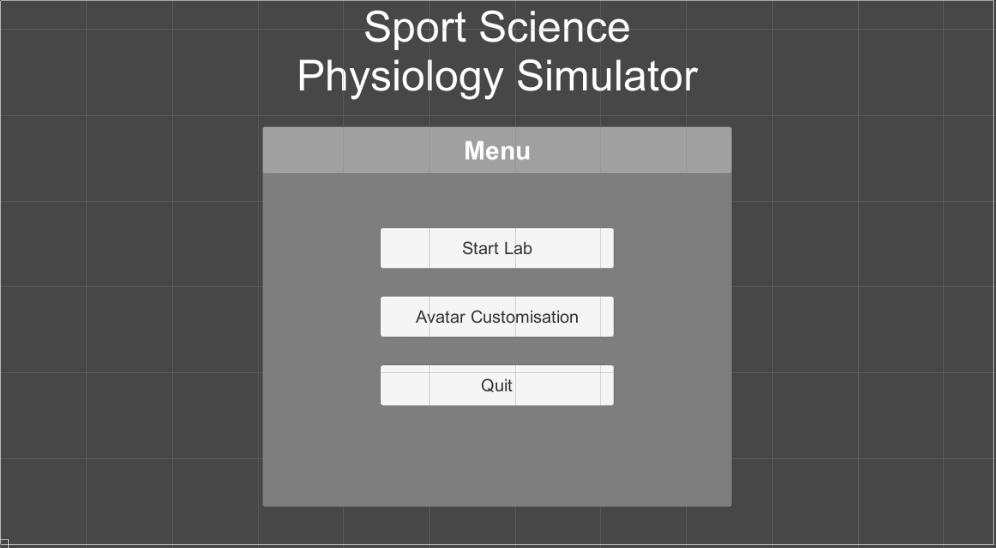
Keyboard - Input data & variables

# Screen Layout

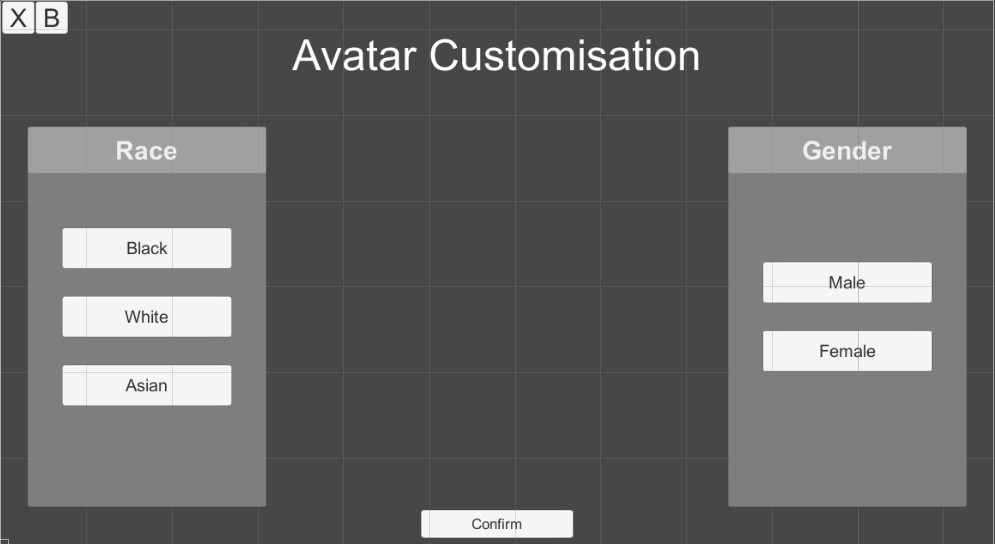
###### First Iteration Main Simulation UI



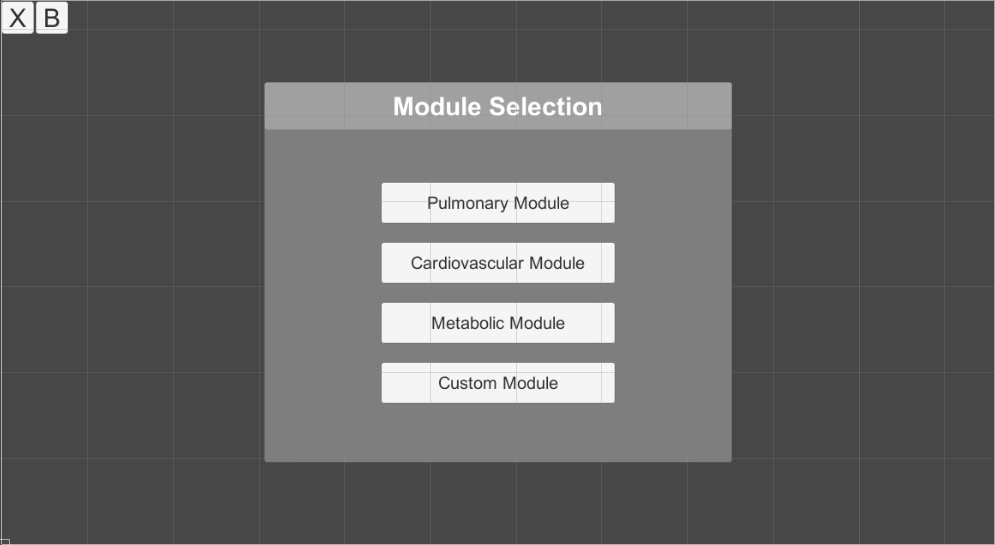
###### First Iteration Start Menu UI

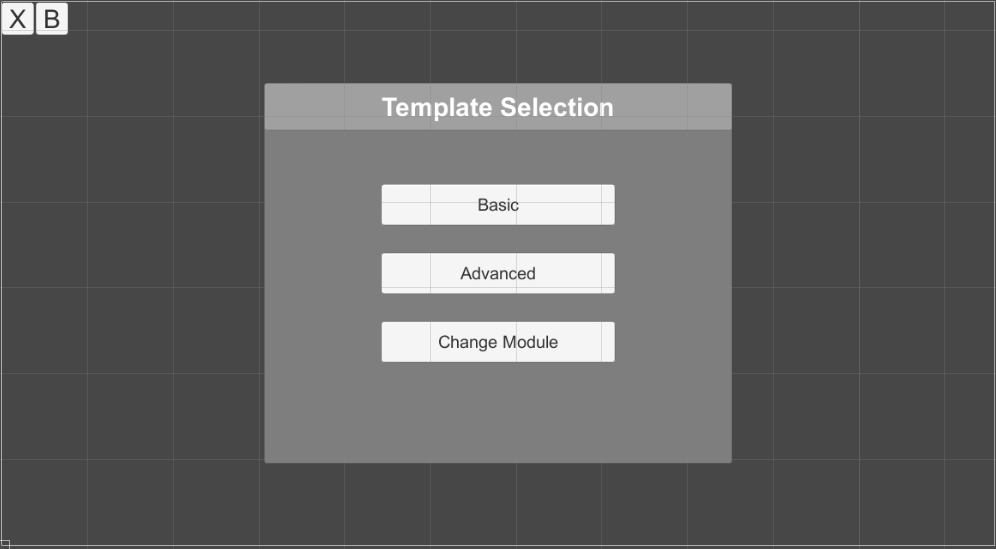


###### First Iteration Character Customisation UI

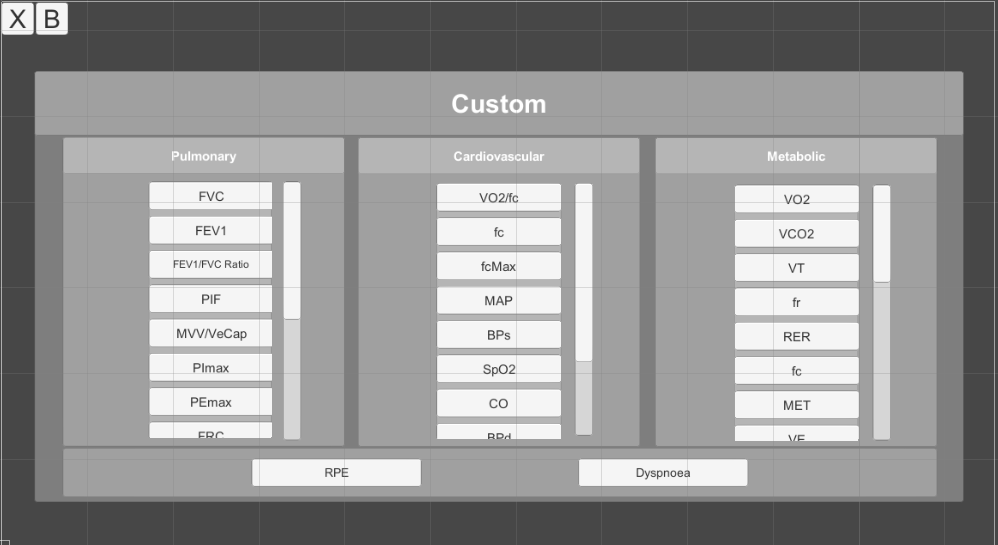


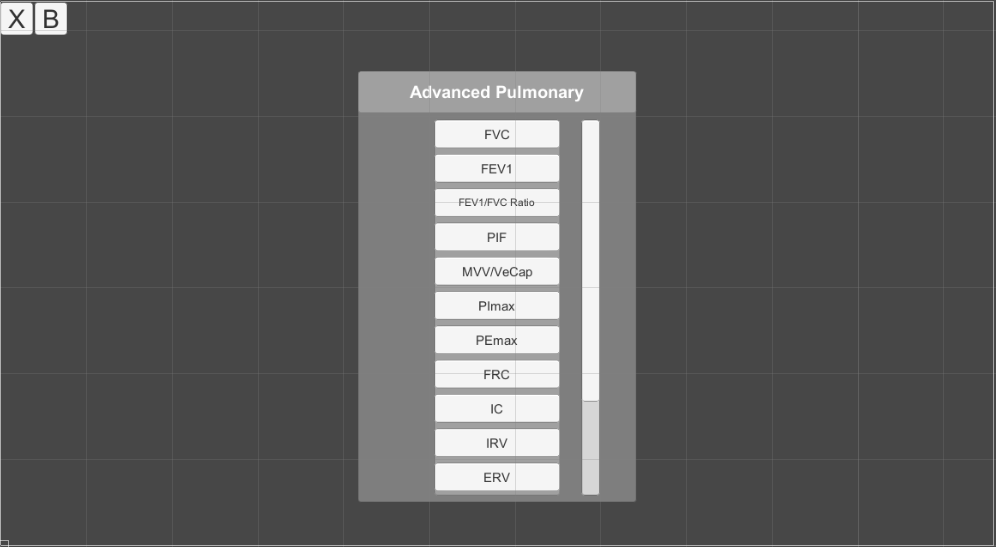
###### First Iteration Module & Template Selection UI



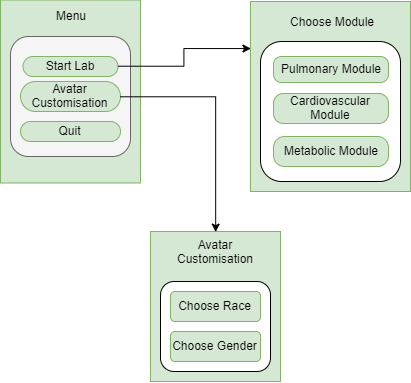


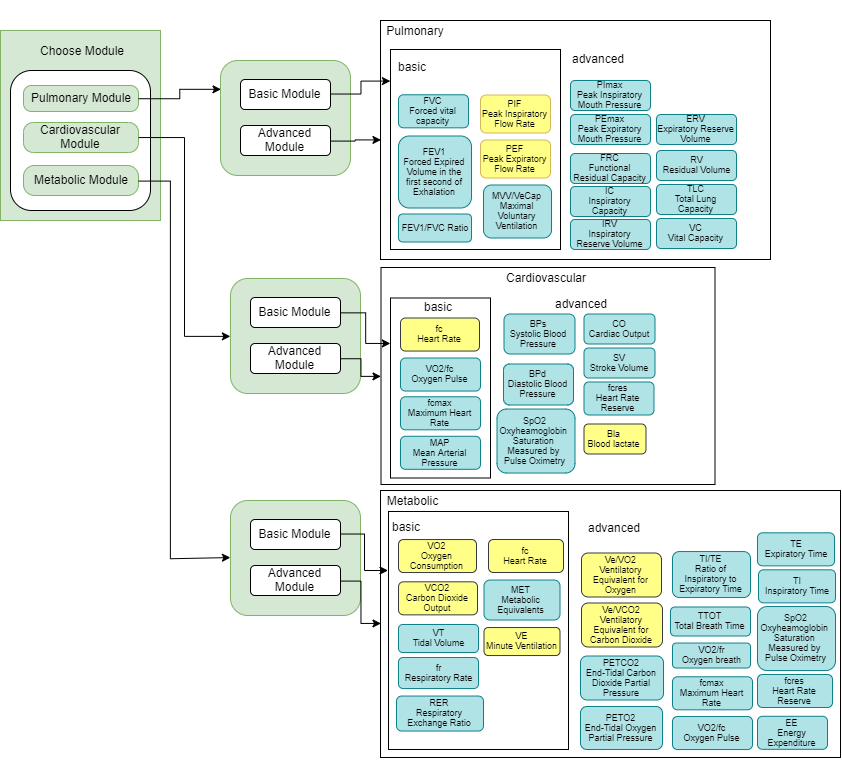
###### First Iteration Variable Selection UI





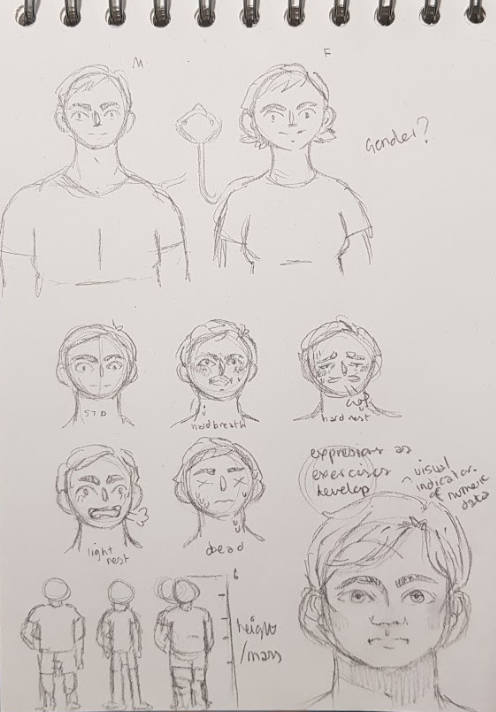
# Walkthrough





# Variables

###### Model Variables

Sex, Race - Options for random/default and manually set

###### Environment Variables

Temperature, Humidity, Oxygen Level - Options for random/default and manually set

###### Equipment Variables

**Bike**

Wattage, Cadence, Duration - Options for default or manually set

**Rower**

Strokes per Minute, Wattage, Duration - Options for default or manually set

**Swimming Flume**

Velocity, Duration - Options for default or manually set

**Treadmill**

Gradient, Speed, Duration - Options for default or manually set

# Art

###### Art Asset List

**Characters:**



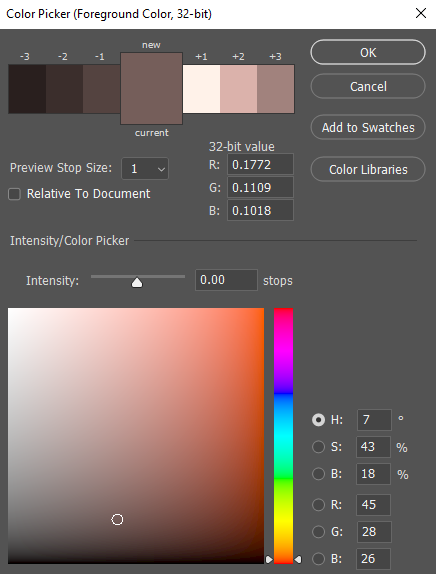
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Equipment:** |  |  |  |  |  |
| alcohol wipes | dynamometer | laser doppler probe | rowing ergometers | t-shape non-rebreathing valve | enviro. control panel |
| anthropometry tape | electrocardiograph | masks (oro-nasal) | skin calipers | thermometers | fans |
| arm crank ergometers | electrical stimulator | mass spectrometer | skin electrodes | thermistors (skin, rectal) | filing cabinets |
| back dynamometer | gas analysers | medical pen | sit & reach box | tissues | first aid kit & defibrillator |
| bacterial filters | gas sampling port | metabolic carts | snorkel | treadmills | freezer |
| blood pressure machine | gauze | metronome | spirometer | weighing scales | ice |
| bottled gas | hand grip dynamometer | mouth pieces | stadiometer | & bioelectrical impedance analysis | medical couch |
| calculator | heart rate watches & transmitter belts | needles | stairmaster | wireless forecast station | phlebotomy chair |
| catheters | hydrostatic weighing | nose clips | stop cock valves | **Background / Building** | sharps bin |
| counters | jump meter (digital) | pulse oximeters | stop watch | bin | tables |
| cycle ergometers (Wingate | lactate & glucose analyser | Q-sweet | syringe (calibration) | chairs | telephone |
| & electronically braked) | lancets | respiratory muscle trainers | syringes (blood) | clock | white board |
| Douglas bags | laptops | respiratory pressure meter | swimming flume | clinical waste bin |  |
| dry gas meters | laser doppler perfusion & temp. monitor | respiratory tubing | tablet e.g.ipad | cupboards for storage |  |

###### Art Style

**** 

**Character Art Style 1**  **Character Art Style 2**   **Character Art Style 3**

Art style 3 was selected. This style is the easiest to replicate for non-art team members as it requires no fine lines and minimal artistic skill, being very forgiving with the “sketchy” lines. Process involves finding a suitable reference image, lowering opacity and tracing the object. Minimal shading (only for the base colour of the asset) for dimension and speed of production, with light coming from the top left angle.. Brush use in lining is “hard round” 100% hardness/opacity in size 4, colour as R45 G28 B26 H7 S43 B18

All characters to use same clothing and colour palettes of the clothing, equipment assets to use shared colour palette for same colours i.e 1 blue and dark blue tone for shading. Colour used to represent black won’t have a darker “shade” tone as this will contrast too highly with the outline, and black will also be used for metallics. No text on any assets - block colour to be used to replicate text position instead to ensure no breach of copyright and easier reproduction for other team members.

# Code

###### Code Implementation Guidelines

* Keep each major function of the code in separate documents, clearly marked, and made public so the other documents can access it.
* Make sure the entire coding team is aware of what you’re doing before starting on a new part of the project, to make sure there are no conflicts.
* Keep the project in the proper folders in your computer, and make sure you are always working on the same project that the source control is checking.
* If you don’t understand how to use the source control, make sure to consult another coder before clicking anything, to make sure you don’t delete anything important.
* Create a pull request at the end of the session, only after successfully testing and saving your work.

###### Code Conventions

* Use the name of the variable the function is calculating in the name of the function
* Always put braces ({}) on separate lines to keep code clean and uniform, unless there is nothing inside of them.
* Always make sure to comment sections of the code that aren’t immediately obvious such as variables or complicated functions.
* Keep variables in brief, succinct lists, and sorted into groups.
* Keep code separated into functions, to make the programme’s function as smooth and modular as possible.

###### Code Comments

* Always add explanations to variables and complicated functions.
* Use CAPITAL LETTERS to note things that need to be changed or added to.
* Do not add so many comments that the quality of the code itself is detracted from.
* Do not add pointless comments, unless they are for your own sake, which should be deleted before pulling code.

###### Implementation of Features

* The programme functions as a calculator, with certain variables being input by the user, and other variables being calculated by the program, with both numerical and visual responses.
* There will need to be two halves of the programme: one that calculates the variables, and understands it, and another half that takes that information and turns it into a visual and numerical response that can be shown to the user.

###### Testing and Validation

* Test code as much as possible, at least once a session.
* Save as much as possible, at least before and after each considerable change to the code, to prevent data problems deleting work.
* A great deal of the information we are working with will be from a sheet given to us by our client; making sure to regularly check and double-check this sheet will ensure no errors are made in copying it to functional code.

###### Modules

* Split into basic, advanced and custom
* Basic modules will just require the output of a few predefined variables that Mitch has already given us. This will be by far the simplest to implement and as such will not be discussed here.
* Advanced modules allow the user to select whichever variables they want to be measured, as long as those variables are within a list given by Mitch. Should be able to drag and drop or use a tick box to see which ones they want implemented.
* Custom modules are fucking bullshit because we have to just be able to use any variable from any of the modules.

###### Implementations

Text boxes - A simple way to display information, difficulty will be in placing them on the screen in the right place depending on the variables. If we have a limit on variables we can display we could do that many boxes and then just assign each variable a number and apply them to the boxes depending on that number. E.g if Tidal Volume is variable 1 it will go into text box 1 and so on.

# Technical Design Limitations

# Sound

# Stress Test

# User Testing & Prototyping

# Comparative Applications

* Labster-Exercise Physiology - <https://www.labster.com/simulations/exercise-physiology/>
* Pulse Physiology Engine - <https://physiology.kitware.com/>
* Virtual Physiology - <http://www.virtual-physiology.com/>
* Bio Gears - <https://www.biogearsengine.com/>

# 

# Technology

###### Art

**Adobe Photoshop**

###### Code

**Unity** - An industry standard software that allows the possibility of progressing the software further into a 3D application if the project is worked on further.

###### Version Control

**GitHub** - It is a free and widely known form of version control that connects well with unity, allows everyone easy access and safely works with the code and art assets.

###### Team Communication

**Trello** - Gives clear visuals of objectives, in different stages of completion, that are viewable by the entire team.

**Discord** - Allows simple communication between a group of people and has the capability of voice chat if needed.

# Production time frame

###### Art

|  |  |  |  |
| --- | --- | --- | --- |
| **Tasks** | | **Time(hours)** | **Dependencies** |
| **Character Art** | | | |
| 1 | Character draft/defining art style | 1-3 |  |
| 2 | Collecting reference images: | 1 |  |
| 3 | Character Assets w/ singular outfit | 4-6 |  |
| 4 | Character Assets w/ both outfits and mask | 3-4 |  |
| 5 | Alternative Facial Expressions per tier | 3-4 |  |
| 6 | Converted Character Assets into PNG | 1 |  |
| **Environment Art** | | | |
| 7 | Collecting reference images | 1 |  |
| 8 | Outline of all red assets | 5-10 |  |
| 9 | Colour and shading of all red assets | 10-15 |  |
| 10 | Converting all red assets to PNG w/ and w/o outline | 1-2 |  |
| 11 | Draft Background | 1 |  |
| 12 | Completed Background | 1-3 |  |
| 13 | Outline of all blue assets |  |  |
| 14 | Colour and shading of all blue assets |  |  |
| 15 | Converting all blue assets to PNG w/ and w/o outline |  |  |
| 16 | Outline of all green assets |  |  |
| 17 | Colour and shading of all green assets |  |  |
| 18 | Converting all green assets to PNG w/ and w/o outline |  |  |

###### Design

|  |  |  |  |
| --- | --- | --- | --- |
| **Tasks** | | **Time(hours)** | **Dependencies** |
| **Design Conceptualisation** | | | |
| 1 | Meet with client to discuss project | 2 |  |
| 2 | Process information from client into application mechanics | 1 - 2 | 1 |
| 3 | Process information from client into potential ui design concepts | 3 | 1 |
| 4 | Determine mechanics specifics | 3 | 2 |
| **Documentation** | | | |
| 5 | (Iterating) Gameflow Chart | 5 | 3, 11 |
| 6 | (Iterating) Game Design Documentation | 6(requires iteration) | 1, 2, 3 |
| **UI Conceptualisation** | | | |
| 7 | Main menu UI | 1 | 3 |
| 8 | Avatar Customisation UI | 1 | 2, 3 |
| 9 | Choose module menu(s) UI | 1 | 3 |
| 10 | Virtual Lab UI | 6 | 2, 3, 4 |
| **User Experience** | | | |
| 11 | Determine user experience (‘gameflow’) | 6 |  |
| 12 | User experience testing | Iterative, 2 | (Programming dependencies) |
| 13 | UI Final Designs | 20 (including implementation) | 7, 8, 9, 10, 11, 5 |

###### Coding

|  |  |  |  |
| --- | --- | --- | --- |
| **Tasks** | | **Time(hours)** | **Dependencies** |
| **Set-up** | | | |
| 1 | Gather information from Client | 2 |  |
| 2 | Understand and conceptualise information from Client | 5 | 1 |
| **Basic Coding** | | | |
| 3 | Create Cardiovascular module | 4 | 1,2 |
| 4 | Create Pulmonary module | 15 | 1,2 |
| 5 | Create Lungs module | 1 | 1,2 |
| 6 | Create Energy module | 2 | 1,2 |
| 7 | Create Exercise module | 2 | 1,2 |
| 8 | Apply created UI to code | 3 | 3,4,5,6,7 |
| 9 | Developer Testing | 2 | 8 |
| **Advanced Coding** | | | |
| 10 | Create Timer module | 3 | 8 |
| 11 | Create Character Customiser module | 4 | 8 |
| 12 | Create Water/Body Heat module | 4 | 1,2 |
| 14 | Create more realistic models of previous modules using the timer | 5 | 3,4,5,6,7,10,12 |
| 15 | Consolidate code to create Functioning Calculator Prototype | 2 | 11,14 |
| 16 | Developer Testing | 2 | 15 |
| **Finalising** | | | |
| 17 | Applying animation and art to the models |  | 15 |
| 18 | Creating visual output that changes with the variables |  | 17,15 |
| 19 | Convincing animated product with numerical and visual output |  | 15,17,18 |
| **Testing** | | | |
| 20 | Developer Testing | 2 | 19 |
| 21 | User Testing | 2 | 19,20 |
| 22 | Client Testing | 3 | 19,20,21 |
| 23 | Debugging | 5 | 19,20,21,22 |
| **Final Product** | | | |
| 24 | Finalising | 5 | 19,23 |
| 25 | Delivering to Client | 1 | 22 |

# References