Steppy Giuseppe Final Report

INFT3970 IT Major project

C3163727 - Emily

C3236400 - Max

C3236401 - Duncan

C3220929 - Sharlene

C3201195 - Simon

C3206003 - Eddie

Executive Summary

Steppy Giuseppe is an infinite platformer video game that is based in the edu-tainment genre. The user plays as Giuseppe, a runner that wishes to explore the world. Players connect their fitbit device and fitbit account to the game and trade in their step count for ingame currency known as ‘Stepcoins’. The purpose and reasoning behind making the game is to inform and educate while at the same time remaining entertaining. This hybrid purpose known as ‘edutainment’ integrates the fitbit step count in order to relate the game back to players, giving purpose and meaning to their real life activity.

The contents of this report will detail our deliverables, technical requirements and a development diary for the course of this major work. A user guide as well as explicitly detailed technical details presented to you.

Table of Contents

Introduction

After identifying a gap in the market for edutainment games that utilize fitbit integration the development team decided to work on this topic for the major project. The objective of the project is to create a fully functional and simple to use web app endless runner game targeted at children aged 8 to 12. The aim of the app is to encourage and reward children to exercise more by rewarding their physical activity in the game. The game will utilise Fitbit activity data (steps) by converting them into in-game currency that the player can use to level up their stats. The more the user exercises, the more in-game coins they will receive, which then allows them to complete levels faster and compete with their friends.

The Current Problem

It is estimated that 28% of all children in Australia are either overweight or obese. Many adults have trouble communicating the importance of health and fitness with their children, and the future implications of an unhealthy lifestyle. Only 50% of schools in the US offer education on nutrition (Youdim, 2016), contributing to the lack of education. Being overweight increases the risk a range of health issues including type 2 diabetes, sleep apnoea and heat intolerance (Health Direct, 2018). According to the Australian Institute of Health and Welfare (2018) the number of obese children are increasing every year, with only 8% of children born between 1974-1977 being obese or overweight.

This project is presented as one of many emerging solutions to this problem.

Outline of the project

Primary Deliverables

The Game

Steppy Giuseppe is a web based application that integrates with a users Fitbit profile data. Users can log into the application using their Fitbit credentials to retrieve daily step activity that is used for the progression of the game. The game will therefore have a login and menu screen. The game will have different levels for the users to attempt.

The Database

The database is used to store step activity, user credentials and Fitbit authentication information such as the access token. Level and skills upgrades to increase the player's speed, stamina and jump height can be viewed and bought by the player. This will be a Microsoft SQL Database deployed on Microsoft Azure.

User Guide

The game is designed to be targeted at children aged 8 to 12 so it is important that game usage is simple and intuitive. This document will contain a basic user guide covering all features.

Website

The user must access the game via the Steppy Giuseppe website. It is hosted using Microsoft Azure and access both the SQL online database and containers for any file storage. The following list shows the main components of the site:

1. Fitbit promotional content

   The user is informed on the site that they must have a fitbit and a fitbit account in order to play the game. Links guiding the user to the fitbit store are therefore on the first page of the site.

2. About Us

   This section shows the members of the Smeeds Gaming development team and also includes the livestream video that was recorded during the presentation of the product.

3. Fitbit Login Authentication

   Before the user can login and add their number of steps into the game, they are redirected to the fitbit site to sign in using their fitbit credentials. These credentials return an access token which the app can use to access and retrieve data stored in the fitbit database.

4. Start Game

   The user can view their previous score in the game, as well as their new score with the added steps from fitbit.

Secondary Deliverables

Livestream recording

The website includes a short recording for promotional and instructional purposes. The film was recorded live during the presentation of the product to a live audience. The film includes a short tutorial that shows the user how to use the fitbit to increase their step count in the game. The instructional video will go for approximately 2-3 minutes.

1. Script

   Interviewer: Hello, welcome to our program this afternoon. Now, have you heard of Exergaming, video games encouraging physical activity? Well right now i present to you SMEEDS Gaming and their game “Steppy Giuseppe”! let’s get a round of applause. **PAUSE FOR APPLAUSE**

   Interviewer: Hi, Who are you Guys and where did SMEEDS come from?

   Interviewee 1: Good afternoon, we are a team consisting of 6 people. Sharlene our Back-end Database Engineer, Max our Front-end Developer, Eddie our Project Manager, Emily our Graphics Designer,Duncan our Lead Game Developer, Simon our Fitbit API Developer. And the first letter of our names combined creates SMEEDS.

   Interviewer: Great, now what sort of game is Steppy Giuseppe?

   Interviewee 2: Well, Steppy Giuseppe is a 3d platformer runner Exergame. It runs as a web application, and it is integrated with Fitbit API. So you need a FitBit account to login to the game and from the account our game gathers the amount of steps for the previous day, puts the steps data into our ingame currency called “StepCoins” and then the game is ready to be played. Now the user depending on how many “StepCoins” they

have the user can upgrade certain attributes for their player. For example, they can increase the speed of the player or add in a double jump.
There are different levels in the game and the aim is to get the fastest time for the level.

Interviewer: Sounds Awesome, Who is your target Market and what is your objective from the game?

Interviewee 1: Steppy Giuseppe is design for children still in primary school or just starting high school and our objective is to encourage fitness with the addition of video gaming entertainment because Physical inactivity is now the fourth leading cause of death in the world and its responsible for 9% of premature mortality.
We did a bit of background research on inactivity and we found that Sedentary/inactivity time is independently associated with increased risk for all-cause mortality, cardiovascular disease incidence and mortality, and type 2 diabetes.
It was estimated that by replacing 30 min of sedentary time per day with an equal amount of light activity, the risk of mortality could be reduced by 14%, whereas replacing sedentary time with moderate-to-vigorous activity gave a 50% risk reduction.

Interviewer: Very interesting, now say i'm playing the game and then i want to to a break and come back to it later. Will my account be saved so i can still keep my bought attributes?

Interviewee 2: Yes it can. When the user exits the game the profile in which they logged into via fitbit saves the data into a database. This includes the users ID, current amount of "StepCoins", the date the user last played and the players current attributes.

Interviewer: Sensational, Can you show us how Steppy Giuseppe works?

Interviewee 1: Yeah sure thing.
LIVE GAMEPLAY
Interviewees talk through the live game play

Interviewer: Wow how good was that! Now in creating this wonderpiece what challenges did you come across and did you overcome them?

Interviewee 1 & 2: None of us has worked with fitbit before. Conducted research on OAuth2, looked at other fitbit integrated games. Problems with the database connection. So we added more resources on working on it.

Technical Requirements

As our game is a web app, the requirements to run it are quite low. The user will need the following:

- A registered Fitbit account
- Internet access
- Up to date modern browser such as chrome/firefox/safari, or any web browser that supports web apps
- We recommend the below devices and minimum specifications
  Windows: Dual core 1.5Ghz Processor, 2 GB RAM, Intel HD Graphics, 1 GB HD space
  Mac OS X
  Mobile

Limitations of the project

While the project aims to provide a full experience from fitbit to the game, there are limitations and exclusions that the project needs to acknowledge in order to not over promise or underdeliver.

- Users must have an active fitbit account in good standing with fitbit.
- The app will be exclusive to Fitbit manufactured devices, other activity trackers will not be included.
- There is no current leaderboard functionality. The player is only able to view their own time it took to complete a level.
- Data that identifies an individual child can not be stored.

User Documentation

Steppy Giuseppe user manual

Steppe Giuseppe is an endless runner cross platformer game designed to incorporate your real life progress of steps into digital currency to spend in game

You play as Giuseppe who wishes to run to the other side of the game in the quickest and most accurate way possible. He is constantly trying to move forward. The levels consist of several objects and obstacles that are designed to make this task harder.

Controls are simple, WASD/Arrow keys to move directionally, Space for jump and that’s it. Avoid obstacles and try to pick up the foods. But beware; Some foods are unhealthy and provide a negative effect on Giuseppe.

1. QUICKSTART GUIDE.

STEP ONE
Log in or register your Steppy Giseppe account by authenticating with fitbit.
The authentication screen must look like this
And be completed in a web browser. Any attempt outside of a third party web browser will fail immediately and ban your individual account.

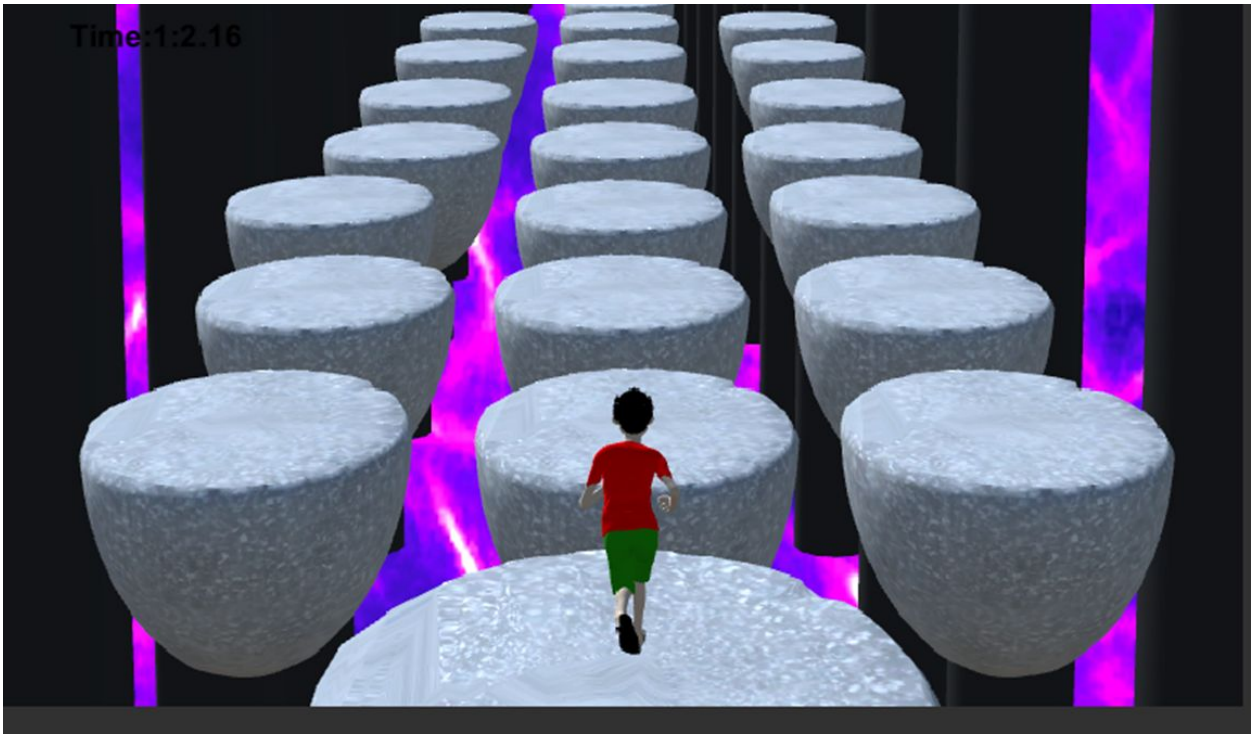
STEP TWO
Check to see that your steps have been recorded and input by the game by observing your step coins. These should be a 1 to 1 match with your step count currently in the fitbit authentication unless you have previously spent them.

STEP THREE
Start the game

2. Main Menu

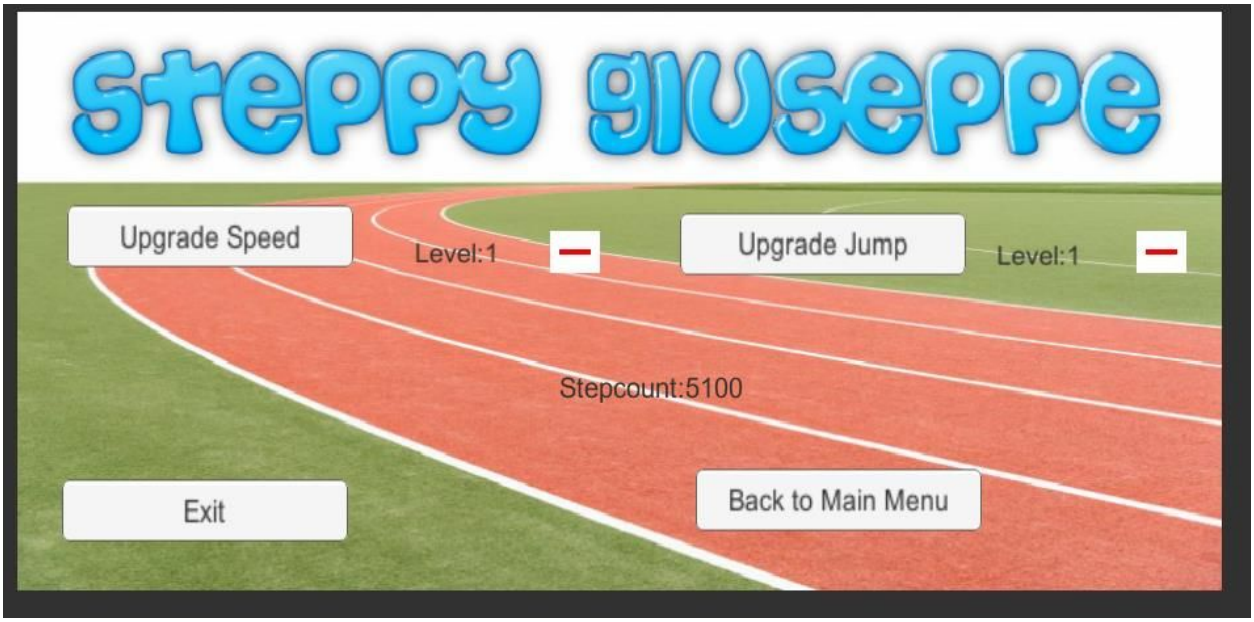
2.1 Play

To begin the game, start by selecting the ‘Play’ button. This will navigate you to the first level of the game. Steppy Giuseppe will automatically use the number of steps retrieved from your fitbit to give you your step coins.



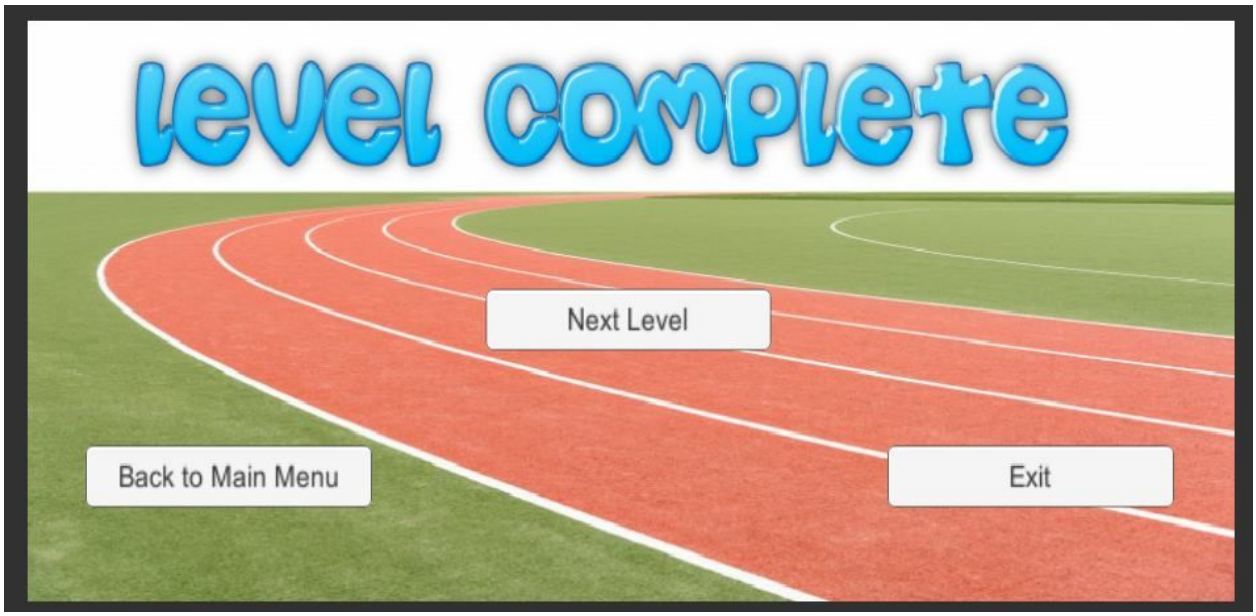
2.2 Skills

The ‘skills’ option will display a list of all the skills on the perk tree that you can acquire as you play through the game. This includes all skills you can acquire for jump, speed and stamina:

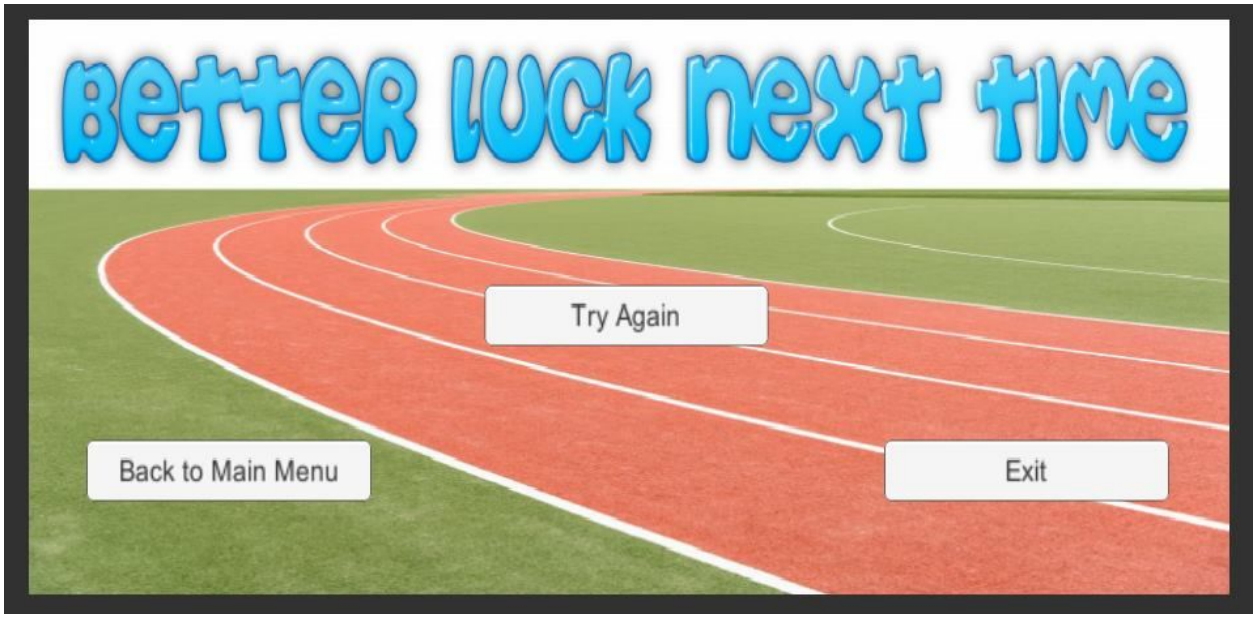


3. Level complete screen

Seen once completing a level, this screen indicates you have progressed through the level all the way to the end, without dying.



4. Level failure screen

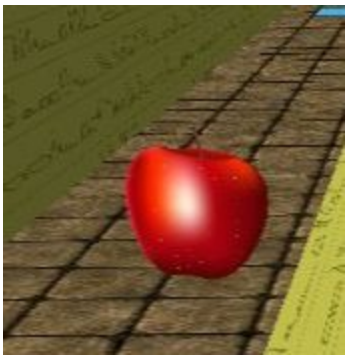


GAME INTRODUCTION

This game is an endless runner that focuses on real world integration through your Fitbit device. While linking your device is entirely optional, the currency in game can only be acquired by taking steps in real life and authenticating them with the game. This currency is known as ‘Stepcoins’ which can be spent on traits such as double jump. Speed and agility increases.

While playing this game you will encounter several obstacles and dangerous environment pieces that will seek to make your time in the world harder to progress. Each of these come with their own unique challenge depending on how many steps you are willing to take in real life to improve your experience in game.

GAME MECHANIC OVERVIEW

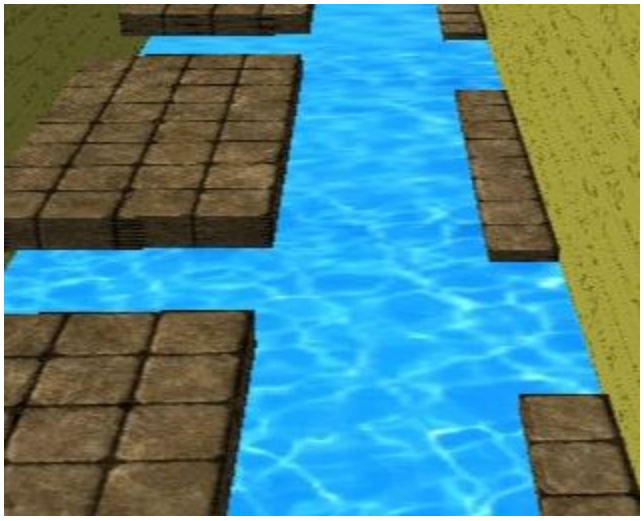


Food and other pickups

These are the lifeblood of Giuseppe. You will want to pick up foods for a speed boost for a short time to help you conquer the obstacles in your path as a runner. These foods range from grapes, apples to negative foods such doughnuts and fries.

Obstacles and hazards.

Obstacles throughout your time in the game will be varied and depend on the level on which you're running.



Platforms and jump hazards

Throughout the game there will be platforms that can only be reached by jumping over other materials such as water, lava or environmental factors. These platforms are a safe haven for Giuseppe.

HOW TO EARN STEPCOINS + TRAIT OVERVIEW

Stepcoins are your ingame currency designed to give a digital value to your real life steps.
At a 1 to 1 ratio step coins are pretty easy to figure out, you take one step; you get one stepcoin. You then spend the stepcoin on traits. Traits go up in price as you progress through levels of them so you'll need to save and accumulate coins over time if you want large perks such as double jump or super speed.

DOUBLE JUMP

This trait allows for the user to jump at the apex of their first initial jump. This can be done once per jump. The double jump trait is useful because it allows for higher jumps on top of a much more efficient Giuseppe. A user with double jump should be able to reliably cross platforms that he would not usually be able to without perfectly timed jumps.

SPEED INCREASE

A flat speed increase for each point added into this trait means Giuseppe is moving faster each time it is ranked up. Being able to run faster in the endless runner allows for the user to finish the game quicker than if someone was running slowly.

Content creation

Basic game mechanics and frameworks

The game functions as an endless runner. We have developed 3 core levels to demonstrate the game in the best way possible with a variation of location/visual aesthetic on each level. Developing the endless runner in unity was accomplished through several trial and error tests, extensive research into the functionality of different aspects such as gravity and velocity of the character. As well as managing the camera in unity so that it didn't fall through the ground.

Pickups

The thought process behind the pickups in game were that they were the 'edu-' part of the 'edutainment' genre. Having various items to pick up with different speed effects programmed to them required users to think about their choices in items they would consume. The items selected and displayed were done with the research of FSANZ nutritional calculator to assess the nutritional value of each pickup.

Having clear and defined choices between a 'positive' or a 'negative' food was a key concept for the game.

Healthy Pickups

Examples of positive foods included:

- Grapes
- Apples
- Bananas

Unhealthy Pickups

Examples of negative foods included:

- Chocolate
- Fries
- Donuts

Stepcoin traits.

Stepcoin traits are the fitbit integration section of the game. Researching into this area of real life purpose into games found that users are more likely to play and care about their progress in a game if their success is determined by factors relevant to their real life. Given that most

games are simply learning machines, building one with legitimate pedagogical purpose required purpose to be given to the data being input.

Gameplay

The levels get more difficult the more the player progresses. The idea is that higher levels have harder obstacles, which forces the player to utilise their levelling progress based on their steps. For example, they may need 3 jumps to reach the next ledge, but they cannot acquire those jumps until they level up.

Platforms

Examples of platform difficulties include:

- In the first level of the prototype, levels are clearer and wider.
- In the final (third) level of the prototype, the platforms can disappear. The player may need to jump more times to get to the next platform.

Obstacles

Examples of obstacles and death hazards:

- Fire
- Snakes
- Spikes

Unity Scripting

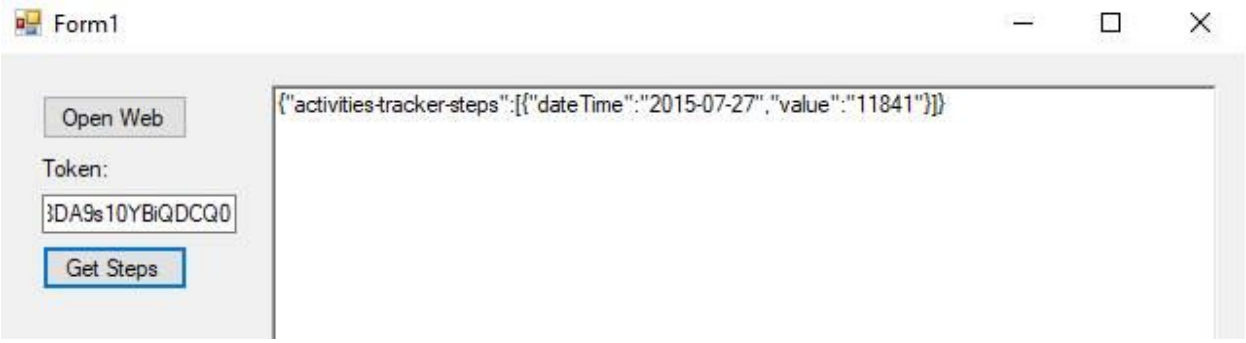
The development team's knowledge base of unity started at an intermediate level. We identified key areas of skill development required in the areas of getting data from a database and integrating this data in the form of a perks/skill tree style, especially setting this data to global and making the data data exist for the whole session as well as some of the advanced physics in the game. These skill developments were made possible through various online coding resources being utilized. Sites such as StackOverflow.com, Unity3d.com, forum.unity and github.com.

Technology/Software

In the development of our project we used several kinds of technology and software. Unity was the main tool used for development of the game as it allowed for us to integrate c# scripts, the authentication for fitbit and develop the game at a core level.

We used visual studio for developing the authentication process. Getting the user activity from the Fitbit Web API required authentication over the OAuth 2.0 protocol. To allow concurrent

development with the game we used visual studio to understand and test the user and app authentication process and retrieve user step data using the C# language. We then transferred the integration procedure from the windows forms application into the Unity game. Use of visual studio was exclusively for implement and testing OAuth and Fitbit Web API.
We used an OAuth client library to simplify the implementation. We did not use any Fitbit client libraries here such as Fitbit .NET.



For the database we used SQL Server Management Studio to write T-SQL scripts to create and insert data into the database. Due to the scope of the assignment, a local database on the PC was used to hold that database. For publishing, it was deployed as a Microsoft Azure SQL Online Database, which the web app connects to in order to retrieve and update data. The main application on Visual studio connected to the database with a connection string in the web.config file.

Developing 3d assets and modelling for various levels we used a SideFx Houdini student license provided by the university.

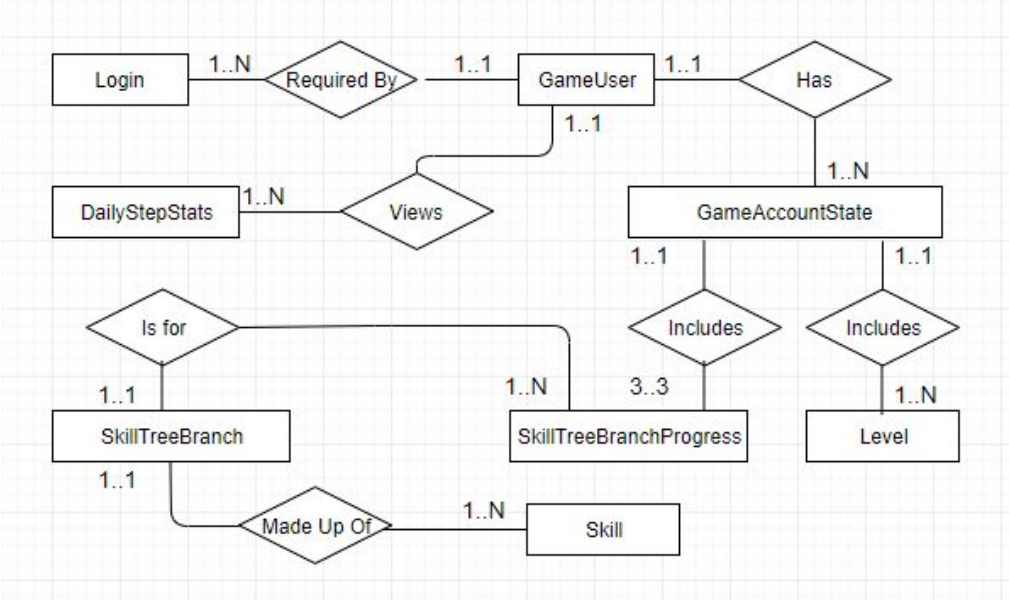
To collaborate and effectively document the process we used google drive and onedrive sharing for various documents.

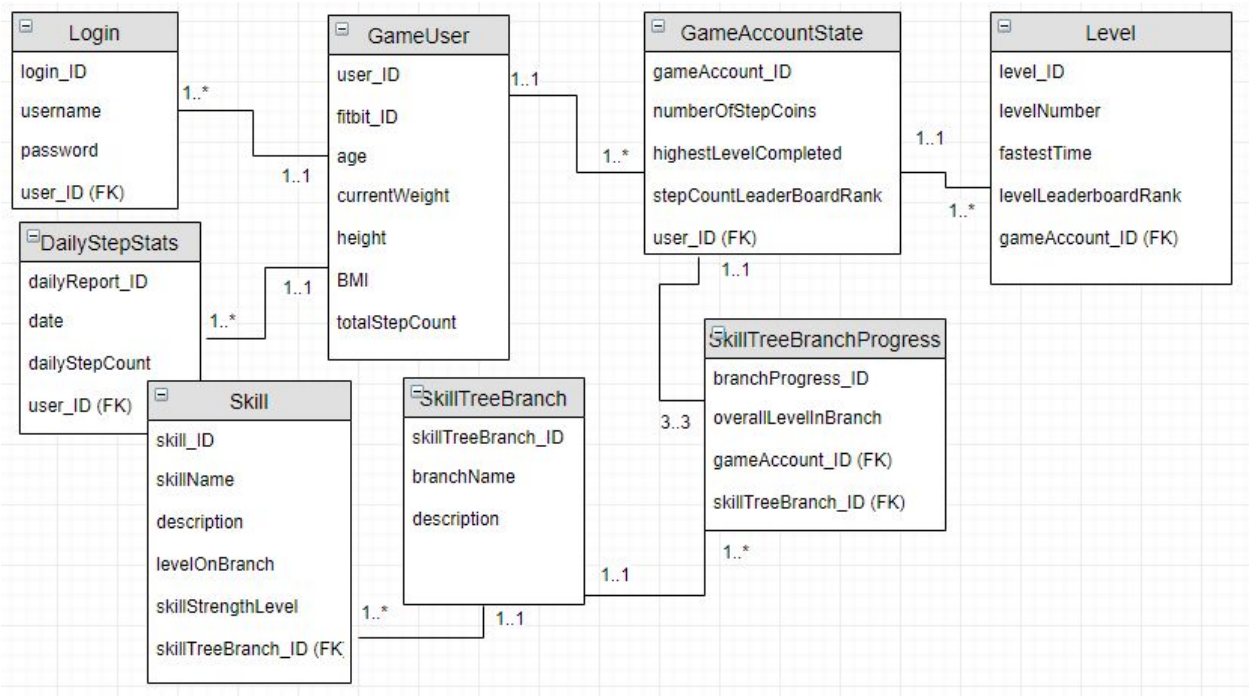
For hosting, we used the MAMP server on a local computer to host the game. The public Azure website then connects to this IP address to run the game.

Technical documentation

Database Documentation

The following are the EER diagrams (using different notations) for the database that was implemented for the project.





The following data dictionary shows information regarding the attributes to be stored in the database:

Entity Types

Entity Name	Description	Aliases	Occurrence
Game User	Stores data about the game user	Player	Can have one or many logins, gamestates or dailystepstats.
Login	Stores login credentials		Can be used by one user.
Game Account State	Stores details about the account, such as their highest level completed.	Account	Associated with 1 user. Includes many levels and 3 skill branches.
Level	User's progress for each level of the game.		Level progress is for one GameStateAccount.
DailyStep Stats	Stores the user's steps daily. The user can see an overview of their steps as progress.		Associated with one user.
SkillTree Branch	The different skill areas the player can upgrade, such as 'speed,' 'jump,' and 'stamina.'		Has many user progresses and levels
Skill	The actual skill on the skill branch, such		Belongs to only one branch

	as ‘jog,’ ‘sprint,’ ‘power run.’		
SkillTree Branch Progress	Tracks which level on the branch the user is currently on.		One progress set it for one user and one branch

Relationships Types

Entity Name	Multiplicity	Relationship	Multiplicity	Entity Name
Login	1..N	Required By	1..1	GameUser
GameUser	1..1	Views	1..N	DailyStepCount
GameUser	1..1	Has	1..N	GameAccountState
GameAccountState	1..1	Includes	1..N	Level
GameAccountState	1..1	Includes	3..3	SkillTreeBranchProgress
SkillTreeBranchProgress	1..N	Is for	1..1	SkillTreeBranch
SkillTreeBranch	1..1	Made up of	1..N	Skill

Attributes

Entity	Attribute	DataType/Length	Nulls	Derived	Default
GameUser	user_ID	CHAR(8)			
	fitbit_ID	CHAR(6)			
	age	INT			
	currentWeight	decimal(4,2)			
	heightCM	decimal(4,2)			
	BMI	decimal(4,2)			
	totalStepCount	INT		Y - From Fitbit data	
Login	login_ID	CHAR(8)			

	username	VARCHAR(20)			
	password	VARCHAR(20)			
	user_ID	CHAR(8)		Y - GameUser Table	
Game Account State	gameAccount_ ID	CHAR(8)			
	numberOfStep Coins	INT			
	highestLevel Completed	INT			
	stepCountLeaderBoardRank	INT			
	token	VARCHAR(100)			
	user_ID	CHAR(8)		Y - GameUser Table	
Level	level_ID	CHAR(8)			
	levelNumber	INT			
	fastestTime	TIME			
	levelLeaderboard Rank	INT			
	gameAccount_ID	CHAR(8)		Y - GameAccount State Table	
DailyStep Stats	dailyReport_ID	CHAR(8)			
	date	DATE			
	dailyStepCount	INT			
	user_ID	CHAR(8)		Y - GameUser Table	
SkillTree Branch	skillTreeBranch_ID	CHAR(8)			
	branchName	VARCHAR(20)			
	description	VARCHAR(50)			
Skill	skill_ID	CHAR(8)			
	skillName	VARCHAR(20)			
	Description	VARCHAR(100)			

	levelOnBranch	INT			
	skillStrengthLevel	DECIMAL(3,1)			
	cost	DECIMAL(7,2)			
	skillTreeBranch_ID	CHAR(8)		Y- SkillTreeBranch Table	
SkillTreeBranchProgress	branchProgressID_ID	CHAR(8)			
	overallLevelInBranch	INT			
	gameAccount_ID	CHAR(8)		Y - GameAccount State Table	
	skillTreeBranch_ID	CHAR(8)		Y - SkillTreeBranch Table	

The schema for the database outlines the structure of the physical database implemented:

GameUser(user_ID, fitbit_ID, age, currentWeight, heightCM, BMI, totalStepCount)
PRIMARY KEY user_ID

Login(login_ID, username, password, user_ID)
PRIMARY KEY login_ID
FOREIGN KEY user_ID **REFERENCES** GameUser(user_ID)
ON UPDATE CASCADE **ON DELETE** NO ACTION

Level(level_ID, levelNumber, fastestTime, levelLeaderboardRank, gameAccount_ID)
PRIMARY KEY level_ID
FOREIGN KEY gameAccount_ID **REFERENCES** GameAccountState(gameAccount_ID)
ON UPDATE CASCADE **ON DELETE** NO ACTION

DailyStepStats(dailyReport_ID, date, dailyStepCount, user_ID)
PRIMARY KEY dailyReport_ID
FOREIGN KEY user_ID **REFERENCES** GameUser(user_ID)
ON UPDATE CASCADE **ON DELETE** NO ACTION

SkillTreeBranch(skillTreeBranch_ID, branchName, description)
PRIMARY KEY skillTreeBranch_ID

Skill(skill_ID, skillName, desciption, levelOnBranch, skillStrengthLevel, cost, skillTreeBranch_ID)
PRIMARY KEY skill_ID
FOREIGN KEY skillTreeBranch_ID **REFERENCES** SkillTreeBranch(skillTreeBranch_ID)
ON UPDATE CASCADE **ON DELETE** NO ACTION

SkillTreeBranchProgress(branchProgressID_ID, overallLevelInBranch, gameAccount_ID, skillTreeBranch_ID)
PRIMARY KEY branchProgressID_ID
FOREIGN KEY gameAccount_ID **REFERENCES** GameAccountState(gameAccount_ID)
ON UPDATE CASCADE **ON DELETE** NO ACTION,
FOREIGN KEY skillTreeBranch_ID **REFERENCES** SkillTreeBranch(skillTreeBranch_ID)
ON UPDATE CASCADE **ON DELETE** NO ACTION

Scope Limitations

It must be noted that the database:

- Has the ability to store data to generate leaderboards
- Allows the player to also view an overall picture of their daily steps history
- Includes more skill branches (other than speed and jump)

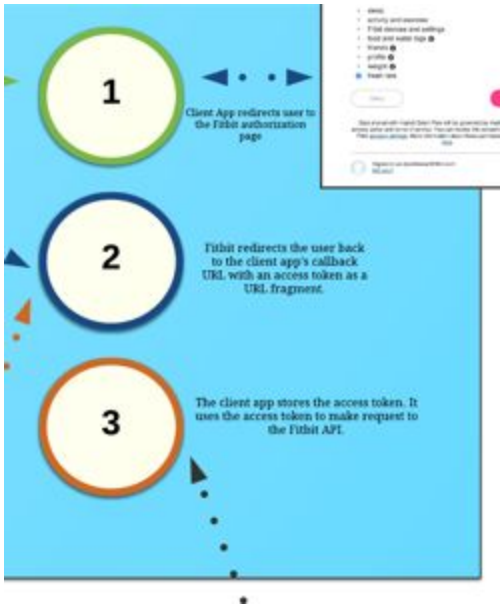
The initial scope for the project included these objectives. However, the scope decreased to not includes these features due to time constraints. Therefore these were not implemented in the actual game prototype. However, these were still included in the database and therefore will make integrating additional features and game mechanics easier in the future.

Fitbit Integration Documentation

The first step of this part of the project was to register an app with Fitbit on their developer portal. For this project the Fitbit app only served the purpose of authenticating the Steppy Giuseppe game and so many settings were filled with dummy information. An official application website was not specified and the callback URL was the local host. The app was registered as a Client app for OAuth 2.0 authentication type, this is instead of a personal app where only the

account holder of the registered app can be authorised to get data, or a web app where Authorization Code Grant Flow must be used.

For the authentication process we chose to use Implicit Grant Flow rather than Authorization Code Grant Flow. The Implicit Grant Flow implementation was most appropriate because it had fewer steps allowing for a more streamlined user experience and was achievable with the use of a database. This also means that the user will not be required to refresh data access tokens.



With our implementation the client side program opens a web browser and directs to the Fitbit websites authorization page. In the link parameters is the permissions we need. We request that the user allow the client side app to read activity data.

Activity data includes variables such as floors, calories, elevations, minutes sedentary and others but our application only queries user steps.

Fitbit users have the functionality to enter their own activity, for example a walk where they did not wear their tracker. Within our web request we have to option to retrieve ‘activity’ or ‘tracker activity’. It is important here that we use ‘tracker activity’ so that we can ensure the steps used for ingame progress are from the tracker and not logged manually.

Once the user has authorized our app to access activity they are redirect to the redirect URL provided in the app settings where the access token is provided as a parameter or the URL. We have configured the redirect URL to take the user to our local host web server were ASP .NET language is used to capture the access token from the address bar and store it in the database. Finally the token and request for data (separate to a request for authorization) are sent to the Web API and the response is captured and stored in a variable.

Unity Game

The unity game includes C# code in scripts to run the game.

Scripts

1. CameraFollow.cs

   This script contains mechanics to make the camera follow the player (when the player chooses to run forward, backwards or side-to-side).

2. CollisionDetection.cs

   Determines whether two objects in the game collide, and if so, what the effects of the collision are.

3. SpeedUpgrade.cs

   Contains mechanics that allows the player to increase their speed or jump levels. This script takes the number of steps stores from the online database and saves it in the game. These steps are then used as the in-game currency.

4. Player.cs & PlayerDeath.cs

   These scripts contain functionality for player movement such as running, jumping and dying. Changes to speed and jump in SpeedUpgrade.cs based on player's steps affect their running and jumping levels.

5. LoadLevel.cs

   When the player accesses the list of levels from the 'Select Level' menu, or if they complete a level, these scripts will load the associated level the user chooses to play.

6. FinishLevel.cs

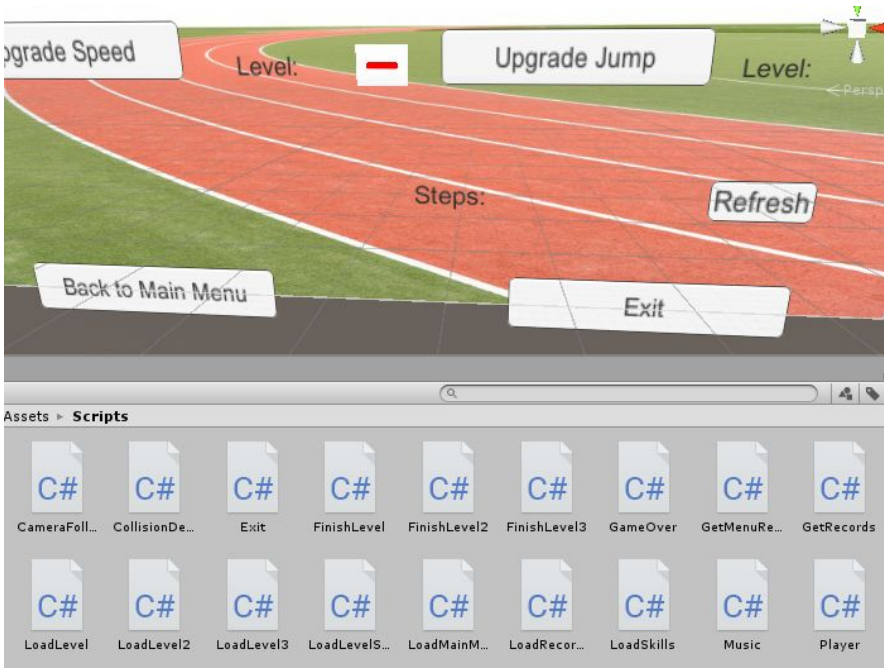
   When the player finishes a level, these scripts ensure that they are directed to a finish page, followed by the main menu.

7. Timer.cs

   This script is called every time the player completes a level and contains functionality to save the player's time. A list of can then be accessed from the Records page on the main menu.

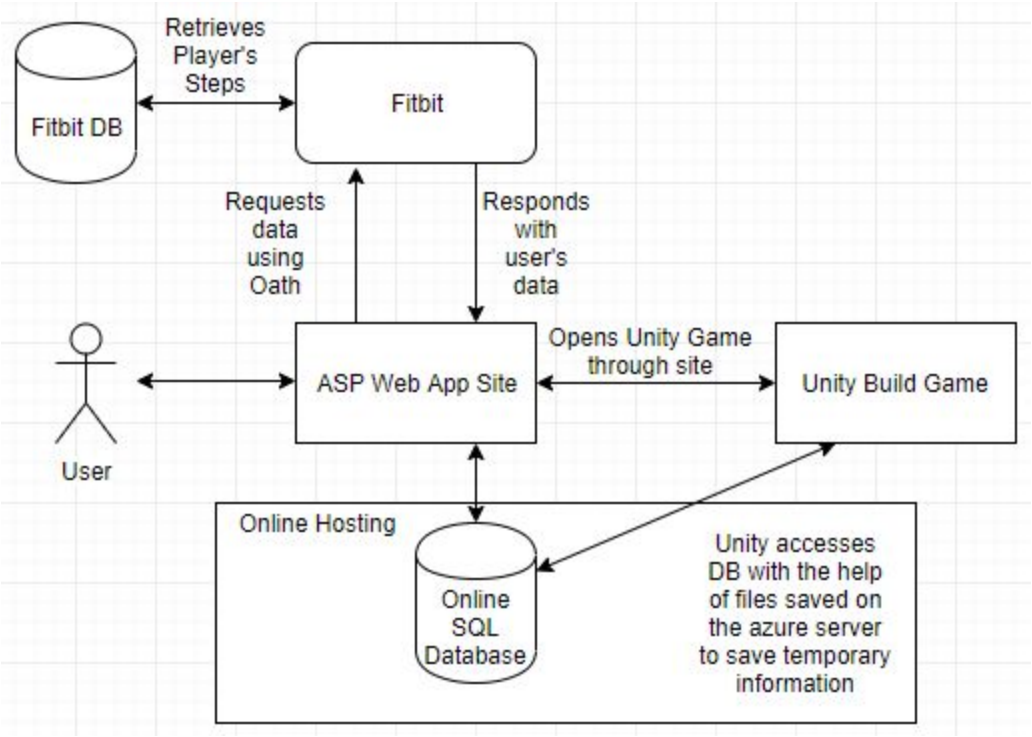
8. getMenuRecords.cs & getRecords.cs

   These scripts navigate to the Records page, where the player can view a list of the times it took to finish a level.



Architecture

Below is a summary of how the different components of the program work together.



Risk management

While the project is digital and it would be perceived to be without risk, there were risk management factors that we considered in the planning, development and implementation phases of the project. While digital projects rarely present a physical risk to users, there are still several risks being taken by utilizing any digital product.

Database security was a risk that we needed to ensure would be minimized. While database security is a huge task, we managed to create a database that would only operate within the parameters specific to our project.

Backups / Data retention - Necessary for any progress based game where we could lose crucial details. Backing up the game physically and digitally as well as the databases was a key risk management measure.

Long playtime - eye strain/fatigue is common in video games where screen time might be extended. Given the endless nature of our game it’s very possible for a user to end up playing for extended periods of time with minimal breaks. Adding a warning as well as a recommended break time

Epilepsy - Given that we have an entire level devoted to a ‘neon’ theme there was a risk factor of an epileptic user encountering severe flashing lights. Adding a warning to the game prior to playing warning users of epilepsy risk.

Fitbit authentication - Fitbit requires users to authenticate their own account through a browser only. Without a browser authentication the fitbit application (ours) will be banned immediately. Fitbit is very particular about the protocol around authentication and the way in which it happens.

Security concerns - While we don’t pull fitbit data that is of major concern for security hazards, the reality is that using the internet in the 21st century with any kind of data that is personally identifiable is somewhat of a concern for anyone. With the chance for data leaks, incorrect obtaining of data and even storage of data that we did not intend for it is essential that we keep a close watchful eye over the data being intaken by the database and the methods of which we use to direct our users to fitbit.

Two factor authentication - Given that users need both a fitbit account that is authenticated and in good standing, as well as a local account in our database it is a two-factor authentication system. Users without the details to a fitbit account can not login on another person's account and as such it is tied to a physical device. IN terms of security, having not just a digital stop for passwords but a physical device (the fitbit) makes password security an easy task.

Ethical concerns and management of privacy.

The main ethical concerns surrounding this particular project was the storing and handling of data that may be considered sensitive for various reasons. Having authentication with fitbit means the project can pull data from the users real life and store specific step count information that the user may not be willing to share with others.

Security of data was a major concern for the project. Privacy in the modern computing era is an increasingly important aspect of any application and this project is no exception. In order to minimise privacy concerns the game only asks for step count authentication from fitbit. This means the only data that is stored to a name is step count.

The age of users as well as underage users means privacy concerns for the data of children. If a child is under the age of 18 the use and storage of their data is governed by a number of different laws depending on which australian state they live in.

Developer diary + Issue log

The development of this project was not without trials and issues. As a development team we faced several obstacles both avoidable and unavoidable. This section of the report outlines the major obstacles with observed consequences/solutions.

Fitbit integration and authentication
Over the course of development we were told extensively by our mentor Geoffrey that the hardest task in terms of technical skill will be integrating fitbit using their API. Undertaking this task meant committing to solving the issues that arose with trying to accomplish this. For 4 weeks we did not give proper respect to this aspect of the project and failed to assign the correct amount of people to it. Having not achieved a working example by week 4 was an oversight of milestones that should have been addressed a little bit earlier as it had potential to seriously bottleneck the project.

Houdini License availability
The licensing for sidefx houdini at our university was limited. Only 20 licenses were available at any one time and the computers that had Houdini would not allow for .hipnc files to be exported through normal methods such as copying to a hard drive or USB.

Unity physics and gravity.
In the early stages of the project, the camera in unity kept falling through the floor and was unable to stay on track with the character through some obstacles.

Terminology and ‘big picture’.
As a development team we could’ve documented our initial scope and vision much more clearly at an earlier date than when we did. While very proactive and prepared, we may have missed a big picture aspect of the project which was nailing down terminology and the core vision of the project very early on. This was most present during consultations where we would often be unaware or undecided on a web app or an executable.

Project schedule
Given that all members of the development team were full time students it was a technical and operational challenge to manage schedules around university to develop this project. Having some group members come from over an hour away from the campus was also an operational challenge that required lots of digital coordination in order to maximise efficiency and effectiveness. The group rose to the occasion for this issue and was able to overcome the challenge to effectiveness by constantly communicating and updating on progress through digital means.

Deployment of 3d modelled assets

One of the largest roadblocks for this game coming together in a timely fashion was the deployment and implementation of 3d assets into our unity session. Assets created in SideFx Houdini were not retaining their textures and skins across platforms and we were losing large amounts of detail + colour. Working around this issue would be porting the assets into unity and reskinning them using unity textures which are blanket colours across all sections which would be timely and very inefficient for the development team. Instead, we explored troubleshooting options such as file format, texture files and even Houdini export settings and found the issue was that we were not exporting the files with their original UV maps included.

Integration of the Database with the Unity game

Initially, the Unity game was only using hard-coded values for the player's speed, jump and stamina attributes. Values for the in-game perks did not change based on the player's number of steps or current level. The database was connected and running with the webapp site and was able to store user's data retrieved from the fitbit database by Weeks 7-8. The next step was to then connect the database to the actual game and read the data for the game. For 2-3 weeks, there was some difficulty trying to get code to connect the unity game to the local database directly.

In week 12, at the weekly meeting with our advisor we informed him that we had not been able to get the database to communicate with the game properly and it was a major problem for our game. This resulted in our advisor warning us that this was a major deliverable for the game and as such our marks would suffer for not being able to deliver a database that works properly with the game.

As a strategy to deal with this issue, we assigned 4 group members to working on database integration. One other group member worked on the final report and documentation. As a development team, we were able to target the issues being brought up with ease. With a tight deadline and time management in final weeks of uni being a major issue we knew we would only have days to address this issue.

Since the database was already connected to the web app, we decided to use those values and find a way to pass them to the unity game. At first, we used helper json files to store temporary data that needed to be pushed into the game from the webapp. These files also allowed us to dynamically transfer information between applications. We then deployed the database online using Microsoft Azure, and hosted the helper files on Azure's servers. These could be accessed directly from both the web app and the Unity scripts using a direct Azure connection to the account that hosted the database.

Readability of Pick-Ups

During the testing period, we discovered that players had difficulty understanding the functions of pick-ups, as the variety of objects lead to confusion. We decided to minimise the number of pickups to four. Given the map variety it would be increasingly harder as more levels were added and speed was increased to make a pickup item readable as negative or positive. The real time choices of users in the testing phase were showing they just couldn't decipher it quickly enough without prior exposure to the pickup and even then, it was more just memory that it was bad rather than a visual queue or idea in their head.

Consistency and readability across the project was improved as a result of testing this feature.

DOUBLE JUMP

The perk that just wouldn't work. Double jump was increasingly more difficult to program as the velocity on top of the height of the jump was hard to program in the endless runner. Given that height and standing was a major issue. With double jump being a major deliverable for the project it was essential that this feature was worked on by multiple people. In a similar fashion to the fitbit authentication problem, we assigned several group members to crowdsource an idea on how to fix the double jump problems being encountered by our programmers.

What have we learned?

This project required several new skills to be developed and old skills to be honed. The major project is no small task and required 6 of us to be dedicated and devoted to the progress of this game over the course of 14 weeks while still maintaining a regular study load.

ONe of the main skills developed and learned over the course of 14 weeks was the teamwork and leadership skills each one of us exhibited. While teamwork in the IT industry is a common requirement from employers very few graduates exemplify or provide instances of where their teamwork skills may be observed. This major project is a testament to the teamwork skills developed.

Using third party application protocol such as the fitbit API was a steep learning curve with several run ins occuring during this process. The fitbit API proved to be far more difficult than originally thought and became a large bottleneck. We solved this issue by dedicating extensive group wide time together to solving this issue and using all 6 people at once for it. This decision, while risky in terms of time management worked for us and allowed for us to complete the module with relatively low impact.

Unity work - extensive skills were developed over the course of this project. Using unity this constantly and for our main concept resulted in having to learn unity concepts that were quite obscure or undocumented. Creating traits such as double jump is largely undocumented on how to accomplish online and we had to trial and error our own examples.

Collaboration as a group was also a challenge for the project. While each team member worked incredibly well together, time and scheduling of each individual member in their final year of university was a logistical challenge. Having some members from the central coast, others from Newcastle meant that group meetings and collaboration required us to digitally work together on key modules, as well as using our time meeting as a full 6 to the maximum efficiency.

Time management over the course of this project was a major concern. Our project took form over 14 weeks of a university semester and required sustained effort the entire time. Management of time came in the form of milestones, reporting progress and two weekly meetings/checkins for the entire group. We managed our time by ensuring all group members had a task to work on at any given time, and a follow up task in the form of assisting another group member once done, or even contributing to the documentation process. Eliminating downtime and cutting time wasting was a core reason for our success as a group.

Web GL - Originally, the game was built without any knowledge of how to deploy or publish it. When we tried to deploy the game, we found that part of the build settings allowed us to deploy it as a web game that opened in the browser. This was called a WebGL build. After watching

some videos on how WebGL works, we also learned that the game would not read files locally when hosted online and therefore we decided to host the files on a server. When deploying the game for the first time, an index.html page was generated which also allowed us to connect the web app and game together. We were therefore able to complete the task through overcoming these obstacles.

References

NESA, *PDHPE - Healthy Eating*. NSW: NSW Education Standards Authority.

Assets References

Andras Nagy, Zsolt Torok- Nature Starter Kit 2. Retrieved From Unity Asset Store

Jeff Johnson-Pyro Particles. Retrieved From Unity Asset Store

Macrobian- Mini Cargo Truck. Retrieved From Unity Asset Store

Michael Kremmel- MK Glow Free. Retrieved From Unity Asset Store

Pngtree- Red Vector Chocolate Bar - https://pngtree.com/freepng/red-packaging-vector-chocolate_801278.html

Clip Art Library- Free Apple Cartoon Pictures. Retrieved from http://clipart-library.com/apple-cartoon-pictures.html

Shutterstock- Dont Walk Sign. Retrieved From https://www.shutterstock.com/search/walk+sign

Clip Art Library- Cartoon Donut Clip Art. Retrieved From http://clipart-library.com/clipart/1122525.htm

Sven Co-op Map Database- egypt1wad. Retrieved from http://scmapdb.com/printer--friendly//wad:tp-egypt-1

Anderson Plywood- Lumber. Retrieved From https://www.andersonplywood.com/lumber/

Painted Furniture Company- Wood Top Colour. Retrieved From https://www.paintedfurnitureco.co.uk/hampton-3-and-4-jumper-chest

Home Depot- Rectange Red Tile Trim. Retrieved From https://www.homedepot.com/b/Flooring-Tile-Tile-Trim/Red/Rectangle/Field/N-5yc1vZargcZ1z0jipuZ1z0jomlZ1z0joom

SafetySign- Basic No Parking Sign. Retrieved From https://www.safetysign.com/basic-no-parking-signs

SafetySign- Park Rules Sign. Retrieved From https://www.safetysign.com/products/4016/park-rules-sign

stockvault - Free Track Stock Photos. Retrieved From https://www.stockvault.net/free-photos/track/?p=9

Fotolia- Background of Athletics Track. Retrieved From https://www.fotolia.com/id/161077831

All Design Creative- Sand Textures. Retrieved From https://www.alldesigncreative.com/3d-textures/sand-texture/

Textures4Photoshop- Cosmic Shockwave Explosion Texture. Retrieved From http://www.textures4photoshop.com/tex/bokeh-and-light/cosmic-shockwave-explosion-texture-overlay-free.aspx

Fotosearch- Clipart of Funny Snake Cartoon. Retrieved From https://www.fotosearch.com/CSP971/k9716800/

Shutterstock- Thanksgiving Cupcake. Retrieved From https://www.google.com.au/imgres?imgurl=https://image.shutterstock.com/image-vector/cartoon-happy-sports-drink-260nw-652231282.jpg&imgrefurl=https://www.shutterstock.com/image-illustration/thanksgiving-cupcake-147017492&h=280&w=260&tbnid=GfzohcPVwdnydM&tbnh=233&tbnw=216&usg=K_5IcbQ3KO0bnfxpbC2u1W4ObEWbM=&hl=en-AU&docid=J3qm3UK5d4i1IM

Cooltext Graphics Generator- Steppy Giuseppe. Retrieved From https://cooltext.com/

Clipart Library - Free Red Stop Sign. Retrieved From http://clipart-library.com/red-stop-light.html

Shutterstock- Tarmac. Retreived From https://www.shutterstock.com/search/tarmac+texture?page=3&section=1&searchterm=tarmac%20texture&language=en

Angels for Peace- Wood Grain texture. Retrieved From https://www.google.com.au/search?hl=en-AU&q=wooden+board+texture&tbm=isch&source=iu&ictx=1&tbs=simg:CAESmAEJu7Hxjv9OzIYajAELEKjU2AQaBggVCAIIBQwLELCMpwgaYQpfCAMSJ9kHyAoCoBWOCdoHhAStFbgWxwreKdco5SfWKOkn6TfqJ-sn1T7tNRow_1qCWT1CcjAtur

79iU6g1iVIDoqkiE0hWQuFsnPolcaUhfPL48SoMH-fcfH2ps7dzIAQMCxCOrv4IGgoKCAgBEgQEwlU1DA&fir=MdH0924Wp7sNiM%253A%252CxxVhrbqlAV4cQM%252C_&usg=AI4_-kQfL2Fi44GMdLeq1aT4jn6GLdOhrw&sa=X&ved=2ahUKEwjxyJC62sPeAhWPb30KHYynDCMQ9QEwAXoECAEQBA#imgrc=jNwJEzO13FJrKM:

Textures4Photoshop- Water Pool Texture Seamless. Retrieved From http://www.textures4photoshop.com/tex/water-and-liquid/water-pool-texture-seamless-and-free.aspx

My Safety Sign- No Smoking in This Area. Retrieved From https://www.mysafetysign.com/no-smoking-in-this-area-warning-sign/sku-s-6146

Coding References

Camera Follow- https://unity3d.com/learn/tutorials/projects/2d-ufo-tutorial/following-player-camera

Scene management- https://docs.unity3d.com/ScriptReference/SceneManagement.SceneManager.html

Find Object for music- https://docs.unity3d.com/ScriptReference/GameObject.FindGameObjectsWithTag.html

Player Movement- https://unity3d.com/learn/tutorials/projects/roll-ball-tutorial/moving-player

WWW Forms- https://docs.unity3d.com/ScriptReference/WWWForm.html

Timer- https://www.youtube.com/watch?v=x-C95TuQtf0&t=154s

Urban by TEMPLATED - templated.co @templatedco
Released for free under the Creative Commons Attribution 3.0 license (Content/license)

How to reference images used. -
Example Name of author. (2018). *Apple, an apple used for a sprite in the game.*[Image]. Retrieved from http://URL